AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Informatyki, Elektroniki i Telekomunikacji
Katedra Informatyki

PROJEKT INŻNIERSKI

Uniwersalny moduł sprzętowego przetwarzania danych oparty na FPGA

Autor: Krzysztof Papciak
Opiekun: dr inż. Jacek Długopolski

Kraków 2016

OŚWIADCZENIE AUTORA PRACY

OŚWIADCZAM, ŚWIADOMY ODPOWIEDZIALNOŚCI KARNEJ ZA POŚWIADCZENIE NIEPRAWDY, ŻE NINIEJSZY PROJEKT WYKONAŁEM OSOBIŚCIE I SAMODZIELNIE W ZAKRESIE OPISYWANYM W DALSZEJ CZĘŚCI DOKUMENTU I ŻE NIE KORZYSTAŁEM ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W DALSZEJ CZĘŚCI DOKUMENTU.

……
PODPIS

Spis treści

1. Cel prac i analiza ryzyka

1. 1. Określenie celów

Celem opisywanego projektu inżynierskiego było zaprojektowanie i wykonanie modułu opartego o programowalny układ FPGA. Urządzenie to miało umożliwić tworzenie własnych sprzętowych implementacji różnego rodzaju procesorów. Moduł powinien wspierać podstawowe układy wejścia/wyjścia do łatwej komunikacji z użytkownikiem i komputerem.

Podstawowym wymaganiem było stworzenie schematu modułu, dobranie odpowiednich elementów oraz zaprojektowanie i wykonanie obwodu drukowanego PCB.

Dodatkowym celem wykonanej pracy było zaprojektowanie i wykonanie przykładowego procesora, prezentującego możliwość platformy. Co za tym idzie, należało także przygotować biblioteki pozwalające na podstawową obsługę peryferiów, takich jak porty wejścia/wyjścia oraz umożliwienie dwukierunkowej komunikacji z użytkownikiem i komputerem.

Efektem wykonanej pracy miał być w pełni funkcjonalny moduł FPGA, który może zostać wykorzystany do implementacji własnych procesorów, jak również tworzenia i uruchamiania innych projektów.

W założeniu należało przygotować port PS2, umożliwiający podpięcie klawiatury oraz wyjście VGA, pozwalające na graficzną wizualizację wyników pracy procesora.

1. 2. Analiza ryzyka

1. 2. 1. Identyfikacja ryzyka

Lp.	Ryzyko (prawdopodobieństwo, 1 - 10)		Skutki (poziom szkód, 1 - 10)	
1	Błąd w projekcie schematu	5	Konieczność wprowadzenia zmian w gotowej płytce PCB lub zrezygnowania z pewnej funkcjonalności	7
2	Błędny dobór elementów	3	Możliwość niewłaściwego działania modułu, w szczególności uszkodzenie elementów	8
3	Wadliwe lub niskiej jakości elementy kluczowe w działaniu modułu	1	Niedziałanie modułu lub jego kluczowych funkcjonalności (konieczność wymiany drogich elementów)	10

4	Wadliwe lub niskiej jakości elementy peryferyjne	7	Niewłaściwe działanie niektórych funkcjonalności modułu (tanie elementy zamienne, długi czas oczekiwania na ponowną dostawę)	4
5	Błąd w połączeniu elementów modułu (np. wykorzystanie niewłaściwych portów układu FPGA)	4	Wolniejsze działanie lub brak możliwości wykorzystania niektórych komponentów	3
6	Błąd w wykonaniu obwodu drukowanego (np. brak lub niewłaściwe połączenie)	2	Ryzyko uszkodzenia lub niepoprawnego działania elementów modułu	8
7	Niewłaściwy dobór parametrów płytki PCB: grubości laminatu, wielkości otworów, odstępów między ścieżkami, wielkości przelotek, grubości ścieżek	4	Mała odporność mechaniczna płytki, zakłócenia, duże opóźnienia w transmisji sygnału, utrudniony montaż elementów	5
8	Zbyt niska jakość układu zasilania (wystąpienie zakłóceń, spadków napięcia pod obciążeniem)	4	Możliwość niewłaściwego działania układów (zwłaszcza FPGA), w najgorszym przypadku ich uszkodzenie.	6
9	Złe rozmieszczenie elementów obwodu drukowanego	4	Problemy ze zmontowaniem zestawu lub jego użytkowaniem	4

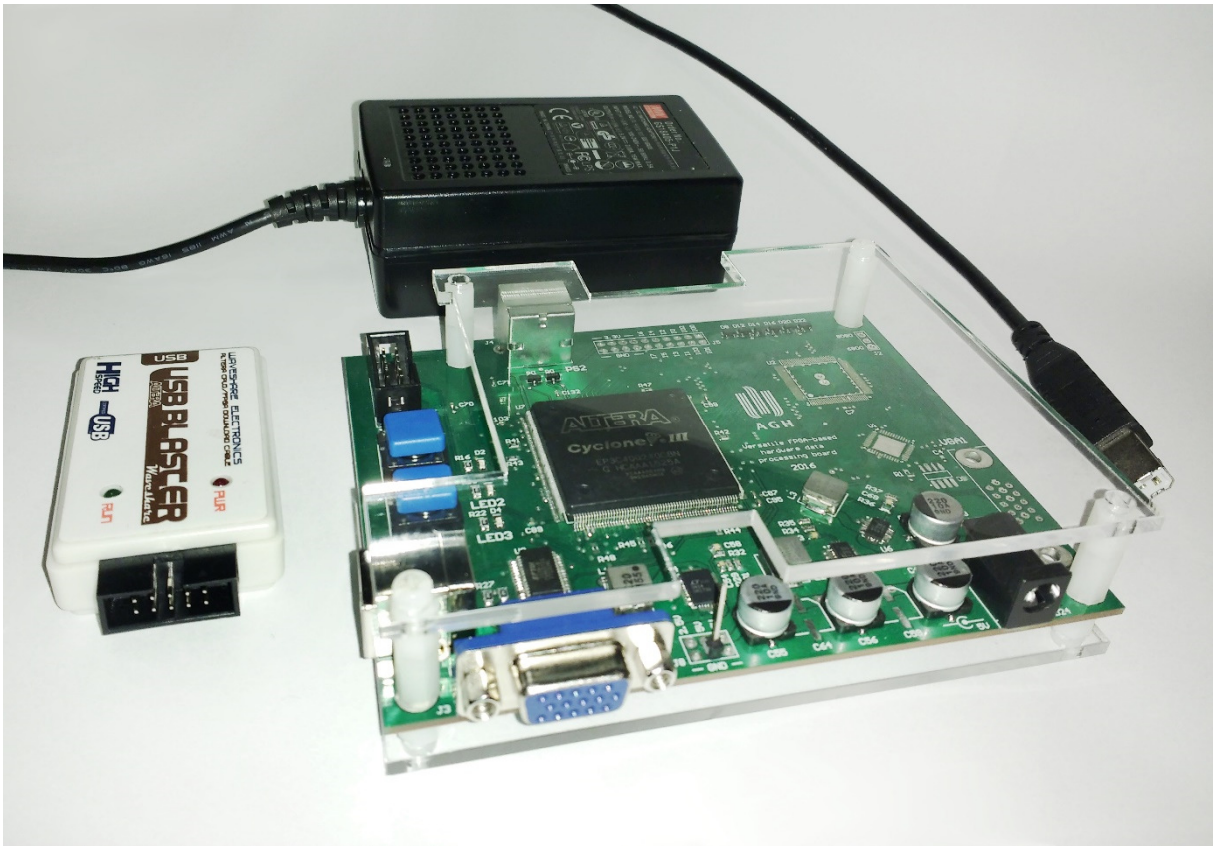
1. 2. 2. Planowane reakcje na wystąpienie ryzyka

Lp.	Ryzyko	Reakcja
1	Błąd w projekcie schematu	W początkowej fazie projektu – poprawienie schematu, po wykonaniu płytki PCB – próba poprawienia błędnych połączeń, w najgorszym przypadku – rezygnacja z pewnej funkcjonalności
2	Błędny dobór elementów	Wymiana elementów lub rezygnacja z funkcjonalności

3	Wadliwe lub niskiej jakości elementy kluczowe w działaniu modułu	Próba rozwiązania problemu z wykorzystaniem obecnych elementów, po niepowodzeniu – wymiana wadliwych elementów na nowe
4	Wadliwe lub niskiej jakości elementy peryferyjne	Wymiana elementów lub rezygnacja z pewnej funkcjonalności
5	Błąd w połączeniu elementów modułu (np. wykorzystanie niewłaściwych portów układu FPGA)	O ile wynik błędu nie wnosi dużych problemów w komunikacji – brak działania
6	Błąd w wykonaniu obwodu drukowanego (np. brak lub niewłaściwe połączenie)	Próba naprawienia błędnego połączenia w istniejącym obwodzie drukowanym, ewentualnie rezygnacja z pewnej funkcjonalności
7	Niewłaściwy dobór parametrów płytki PCB: grubości laminatu, wielkości otworów, odstępów między ścieżkami, wielkości przelotek, grubości ścieżek	O ile będzie to możliwe, próba mechanicznej naprawy (np. rozwiercenie otworu, pogrubienie ścieżki przy użyciu drutu lub cyny)
8	Zbyt niska jakość układu zasilania (wystąpienie zakłóceń, spadków napięcia pod obciążeniem)	W przypadku małych zakłóceń, nie jest potrzebne podejmowanie działania. W przeciwnym przypadku, podjęcie próby poprawy jakości napięcia przy użyciu dodatkowych elementów dyskretnych lub wymiana układów.
9	Złe rozmieszczenie elementów obwodu drukowanego	O ile problem pojawi się przed wykonaniem płytki PCB, należy poprawić rozmieszczenie elementów.

2. Opis zestawu SNF-0 oraz przykładowego procesora graficznego

2. 1. Opis elementów zestawu

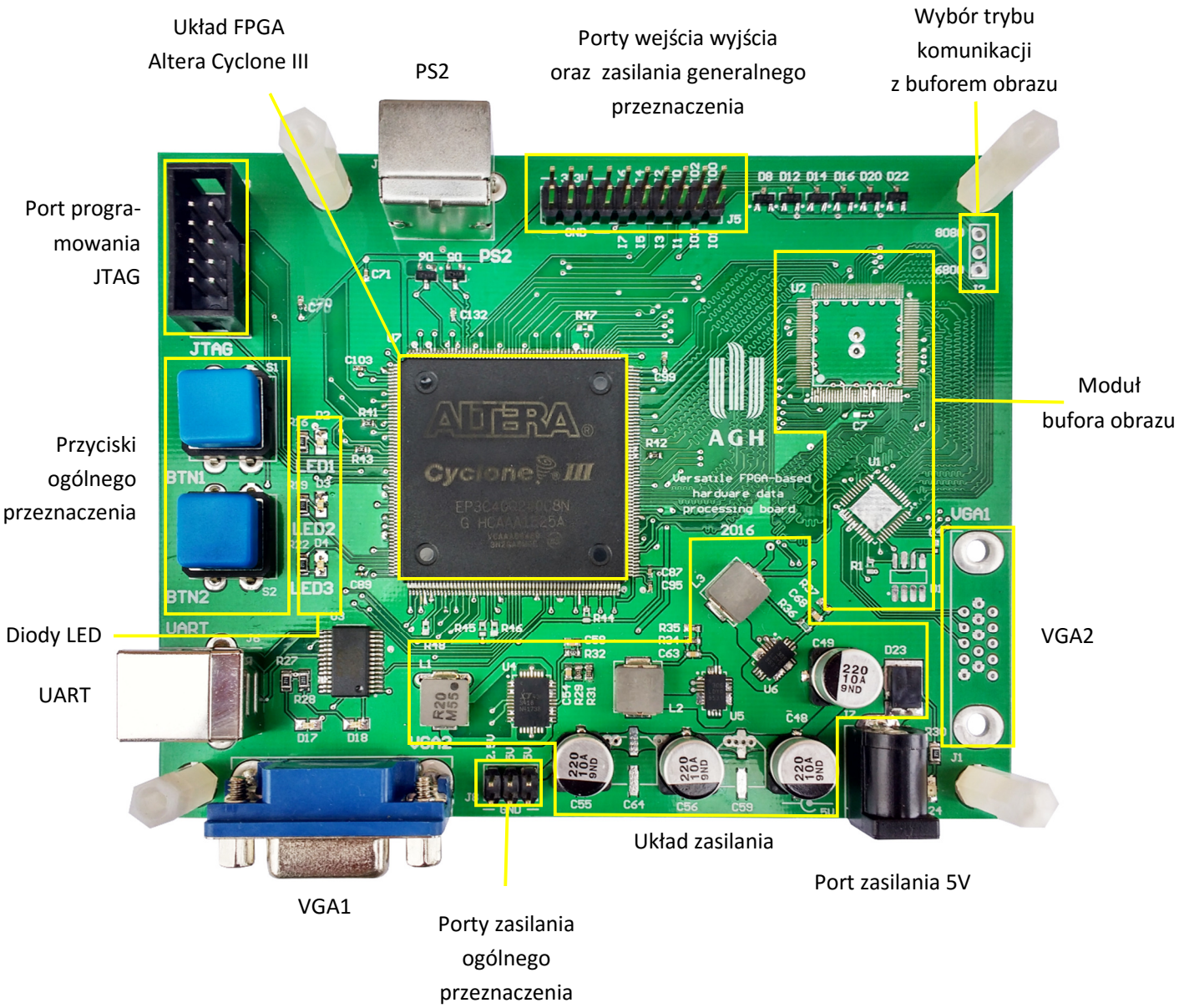


Rys. 1. Elementy zestawu uruchomieniowego SNF-0

Zestaw uruchomieniowy składa się z następujących elementów:

- moduł SNF-0
- programator zgodny z USB-Blaster II
- zasilacz 5V
- kabel USB-B do komunikacji przez port szeregowy

2. 2. Opis elementów płytki PCB modułu SNF-0



Rys. 2. Porty i układy modułu uruchomieniowego SNF-0

Zestaw SNF-0 posiada podstawowe porty wejścia/wyjścia do komunikacji z użytkownikiem i z komputerem. Kontrolki i przyciski pozwalają na proste sterowanie działaniem tworzonych projektów. Porty VGA umożliwiają zwizualizowanie wyników oraz tworzenie projektów generujących grafikę. Jednym z nich jest przykładowy projekt procesora graficznego, wykonany, jako część projektu inżynierskiego. Złącze PS2 zapewnia wygodną obsługę poleceń użytkownika za pomocą klawiatury lub myszki. Porty wejścia/wyjścia generalnego przeznaczenia oraz wyprowadzone napięcia, dostępne na płytce modułu dają możliwość podłączenia zewnętrznych modułów, a także wykorzystanie nowych typów portów komunikacyjnych. Niektóre z układów zostaną wykorzystane w przyszłości do innych projektów, dlatego nie zostały jeszcze zmontowane.

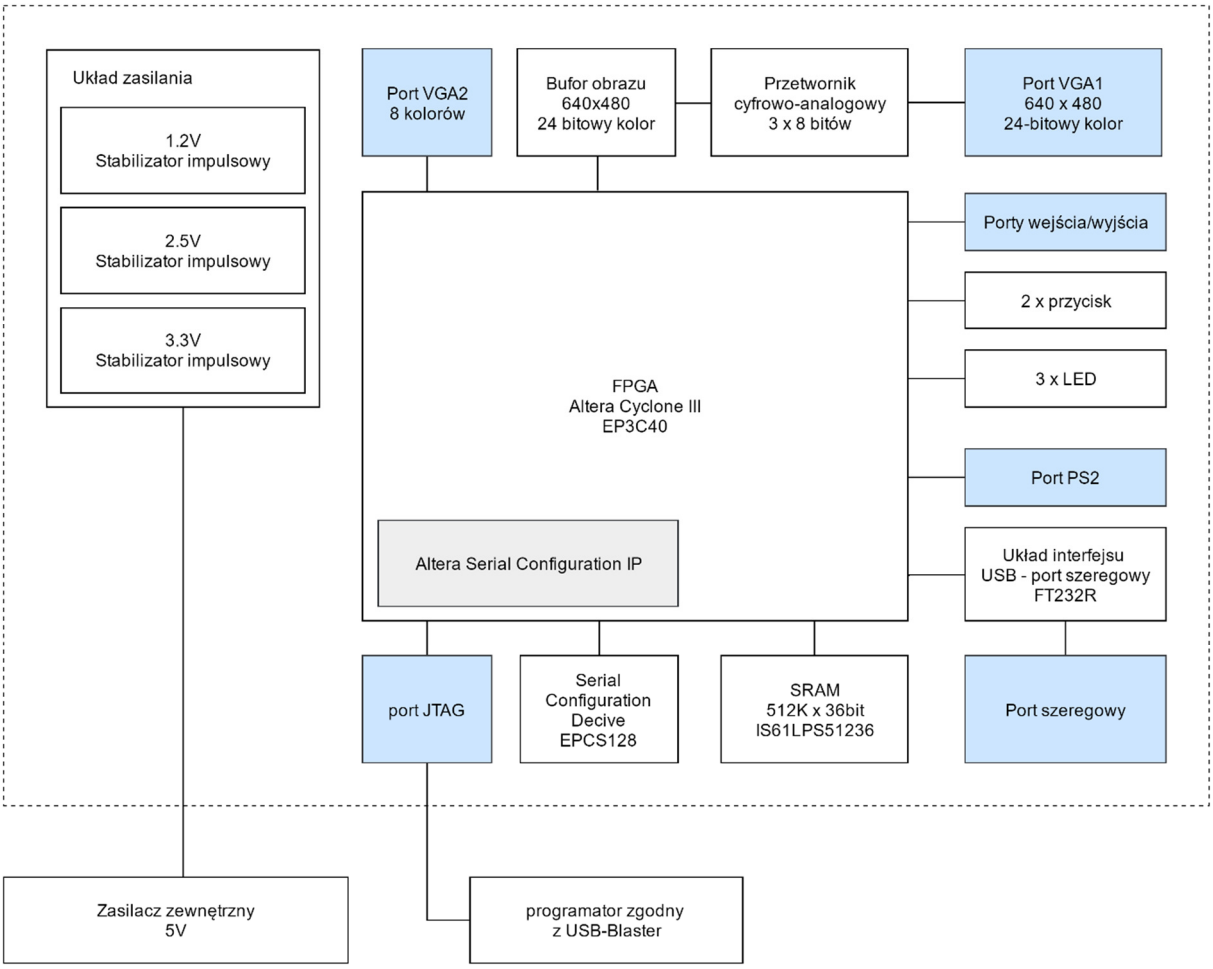
Dostępne porty:

- VGA1 (8 kolorów, dowolna rozdzielczość)
- VGA2 (24-bitowa głębia kolorów, dostępny bufor obrazu, maksymalna rozdzielczość: 640x480)
- PS2 (możliwa obsługa klawiatury i myszki)
- Port szeregowy, zrealizowany za pomocą konwertera z interfejsu USB
- JTAG (port programowania układu FPGA oraz pamięci FLASH przechowującej konfigurację)
- Porty wejścia/wyjścia oraz wejścia ogólnego przeznaczenia zabezpieczone diodami
- Wyprowadzenia dostępnych napięć (3.3V, 2.5V oraz 5V) do zasilania zewnętrznych modułów
- Port zasilania 5V

Układy dostępne na płytce SNF-0:

- Układ FPGA Altera Cyclone III EP3C40
- Pamięć EEPROM konfiguracji układu FPGA (zamiennik EPCS128)
- 2 przyciski ogólnego przeznaczenia
- 3 żółte diody LED
- Oscylator 50MHz podłączony do wejść zegarowych układu FPGA
- Układ interfejsu portu szeregowego do USB
- Bufor i kontroler obrazu SSD1963 (niezamontowany)
- 24-bitowy przetwornik cyfrowo-analogowy dla wyjścia VGA2 (niezamontowany)
- 2-megabajtowa pamięć SRAM (niezamontowana)

2. 3. Schemat blokowy zestawu SNF-0



Rys. 3. Schemat blokowy modułu

2. 5. Opis przykładowego procesora graficznego

Rys. 4. Moduł SNF-0 wyświetlający wyeksportowany model 3D

Specjalizowany procesor graficzny wyświetlający prostą grafikę 3D został stworzony, aby zaprezentować możliwości układu FPGA i portów zewnętrznych.

Częścią składową projektu specjalizowanego procesora graficznego są moduły napisane w języku VHDL, obsługujące wyjście VGA oraz porty PS2 i port szeregowy UART. Można je wykorzystać do tworzenia nowych projektów opartych o moduł SNF-0.

2. 6. Sterowanie przy pomocy klawiatury PS2

Za pomocą klawiatury PS2, można sterować obrotem i skalowanie wczytanej bryły 3D.

- W – zwiększenie kąta obrotu względem osi Y
- S – zmniejszenie kąta obrotu względem osi Y

- A – zwiększenie kąta obrotu względem osi X
- D – zmniejszenie kąta obrotu względem osi X
- Q – zwiększenie kąta obrotu względem osi Z
- E – zmniejszenie kąta obrotu względem osi Z
- Z – zwiększenie skali obiektu
- X – zmniejszenie skali obiektu

3. Wybrane aspekty realizacji projektu

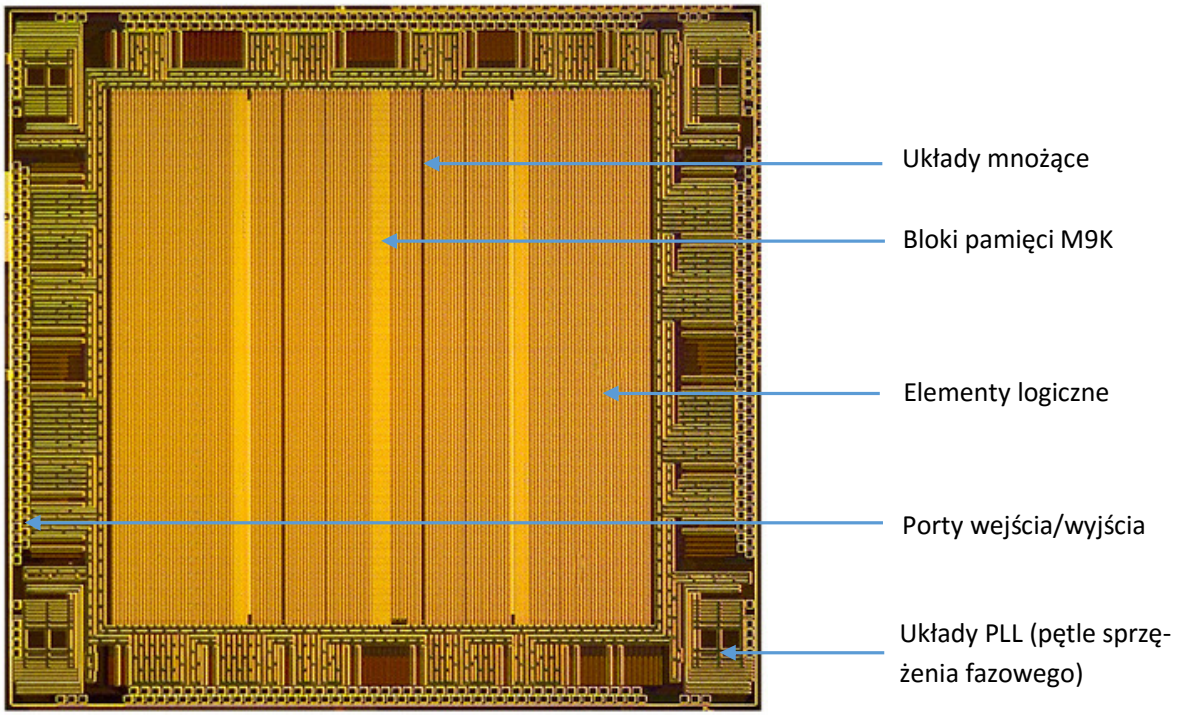
3. 1. Wstęp

FPGA (*field programmable gate array*) jest rodzajem programowalnego układu logicznego, którego głównym podzespołem jest dwuwymiarowa macierz elementów logicznych łączonych ze sobą za pośrednictwem przełącznic. Każda z jednostek logicznych może zostać skonfigurowana do wykonywania prostej operacji, natomiast programowalny przełącznik dostosowany do tworzenia odpowiedniego połączenia między poszczególnymi komórkami logicznymi. Projekt systemu oparty na układzie FPGA może zostać zaimplementowany poprzez specyfikację funkcji każdej z komórek logicznych i selektywną konfigurację programowalnych przełączników.

Projektowanie dużego systemu opartego o układ FPGA jest skomplikowanym procesem, złożonym z wielu zaawansowanych przekształceń, wykorzystujących algorytmy optymalizacyjne. Do tego celu wymagane jest oprogramowanie, które automatyzuje niektóre z tych zadań. Do zaprojektowania opisywanego systemu użyto oprogramowania *Quartus II Web Edition* firmy *Altera*, w wersji 13.1.

Podstawowym elementem budującym układ FPGA jest element logiczny (LE), który zawiera mały konfigurowalny obwód kombinacyjny z przerzutnikiem typu D (flip-flop). Najpopularniejszą metodą implementacji konfigurowalnego obwodu kombinacyjnego jest zastosowanie tzw. *Look-up table* (LUT). Jest to pamięć implementująca prostą funkcję logiczną. Każdemu stanowi *n* wejść przyporządkowany jest odpowiedni stan wyjścia.

Do wykonania projektu inżynierskiego wybrano układ FPGA *Altera Cyclone III*. Strukturę tego układ, z zaznaczeniem ważnych elementów budowy, przedstawiono na rysunku 5.



Rys. 5. Struktura układu FPGA rodziny Cyclone II i opis podstawowych elementów

3. 2. Zastosowanie układu FPGA Altera Cyclone III w projekcie

Układy FPGA pozwalają na łatwe prototypowanie. Dzięki możliwości wielokrotnego przeprogramowania, umożliwiają projektantom tworzenie projektów, które później mogą trafić do produkcji seryjnej, jako specjalizowane układy scalone (*Application Specific Integrated Circuit - ASIC*). W przeciwieństwie do nich układy FPGA pobierają więcej mocy i są wolniejsze.

Dzięki swej specyficznej budowie, układy FPGA pozwalają na projektowanie systemów współbieżnych. Może to znacznie przyspieszyć rozwiązywanie niektórych zadań, które stanowiłyby duży problem dla klasycznych procesorów. Ciągle rosnąca ilość podstawowych elementów logicznych, zwiększanie ilości bloków pamięci i stosowanie specjalnych modułów, takich jak układy mnożące, pozwala na tworzenie coraz bardziej skomplikowanych projektów.

Podstawowym założeniem opisywanego projektu inżynierskiego było stworzenie modułu pozwalającego na tworzenie prototypów układów przetwarzania danych, między innymi specjalistycznych procesorów. Wykorzystanie do tego celu układu FPGA pozwala na łatwe i szybkie tworzenie takich projektów. Możliwe jest tworzenie układów intensywnie wykonujących współbieżne obliczenia, np. procesorów wielordzeniowych lub jednostek generujących grafikę.

Jako główny element modułu SNF-0 wybrano układ FPGA *EP3C40* firmy *Altera* z rodziny *Cyclone III*, który wyposażony jest w 39600 elementów logicznych. Liczba tych jednostek pozwala na tworzenie bardziej skomplikowanych projektów. Do dyspozycji projektanta *Altera* oddała 126 bloków pamięci *M9K*, które łącznie mają pojemność ok 1.16 megabitu. Układ *EP3C40* posiada 4 pętle sprzężenia fazowego (PLL) i 126 18-bitowych jednostek mnożących.

Wybrany układ FPGA znajduje się w 240-nóżkowej obudowie PQFP. Ze względu na stosunkowo duże rozmiary, układ ten ma ósmą (najwolniejszą) klasę szybkości. Pozwala jednak na tworzenie układów logicznych taktowanych z maksymalną częstotliwością ok. 400MHz. Dostępnych jest 128 portów wejścia i wejścia/wyjścia ogólnego przeznaczenia.

3. 3. Projekt schematu modułu SNF-0 opartego o układ FPGA

Głównym elementem modułu SNF-0 jest układ FPGA *Altera Cyclone III*, wyposażony w pamięć EEPROM, mogącą przechowywać jego konfigurację po wyłączeniu zasilania. Do układu *Cyclone III* podłączone są wyprowadzenia portów PS2 oraz portu szeregowego za pośrednictwem konwertera USB. Zestaw wyposażono także w dwa złącza VGA. Jedno z nich posiada bufor obrazu, który pozwala na asynchroniczne generowanie grafiki oraz przetwornik cyfrowo-analogowy zapewniający 24-bitową głębię kolorów. Wszystkie elementy modułu SNF-0 zasilane są przy pomocy trzech stabilizatorów impulsowych podłączonych do napięcia wejściowego 5V.

3. 3. 1. Układ FPGA i jego otoczenie

Układ FPGA modułu SNF-0 programowany jest za pomocą programatora zgodnego z Altera USB-Blaster. Konfiguracja wykonywana jest w trybie JTAG. Domyślnie konfiguracja układu FPGA zanika po wyłączeniu zasilania. Zastosowano układ szeregowej pamięci EEPROM, który może ją przechowywać. Po zastosowaniu odpowiedniego modułu IP dostępnego z oprogramowania Quartus II, konfiguracja układu FPGA zostaje odtworzona z pamięci po uruchomieniu zasilania.

Do generowania sygnału zegarowego, podłączonego do wejść układu FPGA, zastosowano oscylator MEMS, który na wyjściu tworzy przebieg sinusoidalny o częstotliwości 50MHz.

3. 3. 2. Pamięć SSRAM

Pamięć SSRAM (*Synchronous Static Random Access Memory*) jest podłączona do układu FPGA poprzez specjalizowany interfejs pamięci zewnętrznej, który udostępnia rodzina FPGA *Cyclone III* firmy *Altera*.

Układ ten komunikuje się z układem FPGA używając 19-bitowej magistrali adresowej i 36-bitowej szyny danych. Dodatkowo wykorzystane są sygnały sterujące umożliwiające użycie *Burst Mode*, w którym dane przesyłane są pakietami, co znacznie przyspiesza komunikację. Sygnał zegarowy pamięci generowany jest bezpośrednio przez układ FPGA. Pozwala to na jego dowolne dostosowywanie i synchronizację transmisji danych. Układ pamięci *IS61LPS51236* użyty w projekcie może pracować z maksymalną częstotliwością 200MHz.

3. 3. 3. Port szeregowy

Podstawowym portem komunikacyjnym wykorzystywanym w module uruchomieniowym SNF-0 jest port szeregowy zgodny ze standardem *RS-232*, podłączany do komputera za pomocą konwertera

z portu USB. Zastosowanie tego sposobu komunikacji pozwala standardowo na osiągnięcie prędkości transmisji na poziomie ok. 10.5 KB/s, co powinno wystarczyć do realizacji podstawowych projektów.

Układ *FT232R* firmy *FTDI Chips* jest zintegrowanym interfejsem USB do portu szeregowego, który oprócz podstawowej funkcjonalności, posiada także między innymi zintegrowaną pamięć EEPROM, generator kwarcowy i stabilizator o niskim spadku napięcia.

Układ firmy *FTDI Chips* oferuje także 5 konfigurowalnych portów ogólnego przeznaczenia. W opisywanej aplikacji podłączone są do diod LED, które wskazują trwającą transmisję. Interfejs USB do portu szeregowego oraz kontrolki zasilane są bezpośrednio z portu USB przy użyciu wewnętrznego stabilizatora, który obniża napięcie z 5V do 3.3V. Wykorzystano proste zabezpieczenie przy użyciu diod, które redukują skoki napięcia w razie przepięć elektrostatycznych lub pojawienia się zbyt wysokiego lub ujemnego napięcia na portach transmisji danych standardu USB.

Ze względu na dużą integrację, układ firmy *FTDI Chips* pozwala na łatwe wykorzystanie, jako konwerter USB – port szeregowy. Dzięki niemu nie jest potrzebne użycie standardowego gniazda RS-232, które we współczesnych komputerach jest rzadkością. Zbędna staje się także konwersja poziomów logicznych, która byłaby potrzebna przy zastosowaniu klasycznego rozwiązania.

Układ *FT232R* wymaga zainstalowania sterowników, które można pobrać ze strony producenta. Jest tam także dostępne oprogramowanie, które pozwala na konfigurację ustawień układu, zapisanych w pamięci EEPROM. Możliwe jest między innymi ustawienie prezentowanej nazwy interfejsu, czy też konfiguracja portów ogólnego przeznaczenia.

3. 3. 4. Port PS2

Podłączenie portu PS2 do układu FPGA zostało zabezpieczone przed skokami napięcia za pomocą diod. Zastosowania tego portu umożliwia łatwe podłączenie do modułu SNF-0 klawiatury lub myszki zgodnej ze standardem PS2.

3. 3. 5. Przyciski i diody LED

Elementami zestawu SNF-0 są 3 diody LED oraz 2 przyciski, posiadające prosty filtr redukujący niepożądane efekty związane z drganiem styków. Zarówno diody, jak i przyciski mogą zostać wykorzystane w projektach w dowolny sposób, pozwalając na interakcję z użytkownikiem.

3. 3. 6. Porty wejścia/wyjścia ogólnego przeznaczenia i wyprowadzenia napięć zasilających

Dostępne są 4 porty wejścia/wyjścia i 8 portów wejściowych ogólnego przeznaczenia. Wyprowadzono także 4 napięcia zasilania o wartości 3.3V, dwa o wartości 5V i jedno 2.5V. Porty zostały zabezpieczone diodami chroniącymi układ FPGA przed skokami napięcia.

Wyprowadzenia wejścia/wyjścia, podłączone bezpośrednio do układu FPGA, mogą służyć do komunikacji z zewnętrznymi modułami. Wejścia mogą zostać także wykorzystane do dostarczania sygnału zegarowego do taktowania logiki zbudowanej na układzie FPGA.

Dzięki stosunkowo dużej szybkości kontrolera wejścia/wyjścia w układzie *Altera Cyclone III*, możliwe jest używanie dostępnych wyprowadzeń, jako portów sygnałowych standardu SPI lub podłączenie do nich zewnętrznego modułu Ethernet.

W chwili obecnej do komunikacji w module SNF-0 dostępny jest port szeregowy, którego szybkość transmisji jest niewielka. Dzięki wykorzystaniu portów wejścia/wyjścia możliwe będzie w przyszłości rozbudowanie podsystemu komunikacji.

Wyprowadzono porty napięć dostępnych na płytce obwodu drukowanego modułu SNF-0, które można będzie wykorzystać do zasilania zewnętrznych podzespołów i modułów.

3. 3. 7. Wyjście VGA2

Wyjście wideo *VGA2* posiada podstawową funkcjonalność wyświetlania obrazu w 8 kolorach. Jego zaletą jest jednak łatwość w użyciu i możliwość zastosowania praktycznie dowolnej rozdzielczości, której częstotliwość generowania pikseli będzie w stanie obsłużyć układ FPGA. Port *VGA2* posiada 3 jednobitowe sygnały kolorów oraz sygnały synchronizacji poziomej i pionowej.

3. 3. 8. Wyjście VGA1, przetwornik cyfrowo analogowy i bufor obrazu

Wyjście VGA1 zostało wyposażone w dodatkowe układy, pozwalające na uzyskanie rozszerzonych funkcjonalności. Dzięki zastosowaniu bufora obrazu możliwe jest przesyłanie danych do wyświetlenia w wybranej kolejności lub też selektywne nadpisanie już obecnych wartości koloru. Pamięć obrazu pozwala także na aktualizowanie danych z dowolną częstotliwością, co umożliwia generowanie złożonej grafiki. Nie byłoby możliwe wyświetlenie obrazu o takiej ilości kolorów i rozdzielczości przy wykorzystaniu bezpośredniego podłączenia portu VGA do układu FPGA, która dostępna jest przy zastosowaniu zewnętrznego układu.

Zastosowano układ SSD1963 firmy Solomon Systech, który jest buforem obrazu o pojemności 1215KB ze zintegrowanym kontrolerem wyświetlania. Dostępna pamięć pozwala na przechowanie obrazu o rozdzielczości do 864 x 480 pikseli przy 24-bitowej głębi koloru. Układ został wyposażony w równoległy interfejs komunikacyjny, wspierający 2 tryby (tryb 6800 i 8080) i różne szerokości magistrali danych (8, 9, 16 lub 24-bitów).

Oprócz podstawowej funkcjonalności, omawiany element posiada kilka przydatnych funkcji pozwalających na manipulację wyświetlanym obrazem takich, jak: możliwość manipulowania jasnością, kontrastem i nasyceniem, obrót obrazu oraz odbicie lustrzane.

Układ SSD1963 został skonstruowany, jako kontroler obrazu dla wyświetlaczy LCD nieposiadających własnej pamięci RAM. Pomimo tego faktu, interfejs sterownika firmy Solomon Systech nadaje się do wykorzystania z wyjściem VGA. Sygnały komunikacyjne w wyświetlaczach ciekłokrystalicznych, które wspiera układ SSD1963 są zgodne z tymi używanymi w standardzie VGA.

Dzięki wbudowanej pętli sprzężenia fazowego (PLL) możliwe jest ustawienie odpowiedniej częstotliwości zmiany kolorów kolejnych pikseli oraz parametrów licznika synchronizacji pionowej i poziomej. Konfiguracja układu odbywa się za pomocą dostępnych interfejsów poprzez wydawanie odpowiednich poleceń i ustawianie wymaganych wartości wbudowanych rejestrów.

Kolejnym ważnym elementem podsystemu wyświetlania VGA1 jest układ 3-kanałowego przetwornika cyfrowo-analogowego ADV7125 firmy Analog Devices. Jest to specjalizowany konwerter przeznaczony do zastosowań związanych z wyświetlaniem obrazu o trzech składowych koloru.

Dla każdego kanału dostępna jest 8-bitowa magistrala wejściowa, przez którą przesyłane są dane o kolorze. Na każdym z trzech wyjść analogowych, podpiętych do złącza VGA, pojawia się sygnał o odpowiednim poziome napięcia, który steruje jasnością aktualnie wyświetlanego subpiksela.

Wykorzystano zewnętrzny układ napięcia referencyjnego 1.235V, którego głównym elementem jest dioda Zenera o dużej dokładności.

Podsystem wyświetlania obrazu ze złączem VGA1 pozwala na tworzenie projektów wyświetlających zaawansowaną grafikę w 24-bitowej głębi kolorów. Wadą tego rozwiązania jest jednak maksymalna rozdzielczość możliwa do uzyskania, wynosząca 640 x 480 pikseli.

3. 3. 9. Moduł zasilający

Aby spełnić wymagania napięciowe i prądowe układów modułu SNF-0, zaprojektowano układ zasilania składający się z trzech stabilizatorów impulsowych, które obniżają napięcie wejściowe 5V do wartości użytecznych dla elementów modułu.

Złącze zasilające połączone jest bezpośrednio z diodą Schottky'ego (posiadającą mały spadek napięcia w kierunku przewodzenia), która zabezpiecza przed podłączeniem napięcia o odwrotnej biegunowości.

Do zbudowania podsystemu zasilania wykorzystano stabilizatory impulsowe firmy *Linear Technology*, charakteryzujące się między innymi niskim spadkiem napięcia pod obciążeniem i dużą wydajnością prądową. Niebagatelnym warunkiem wyboru stabilizatorów impulsowych *LTC3616* i *LTC3418* była mała ilość elementów dyskretnych potrzebnych do ich uruchomienia i łatwa dostępność.

Układy stabilizujące napięcie użyte w projekcie posiadają wbudowane tranzystory kluczujące. Do ich poprawnego działania potrzebne są tylko kondensatory, cewka i rezystory konfigurujące wartość napięcia wyjściowego i częstotliwość pracy układu.

3. 4. Projekt obwodu drukowanego

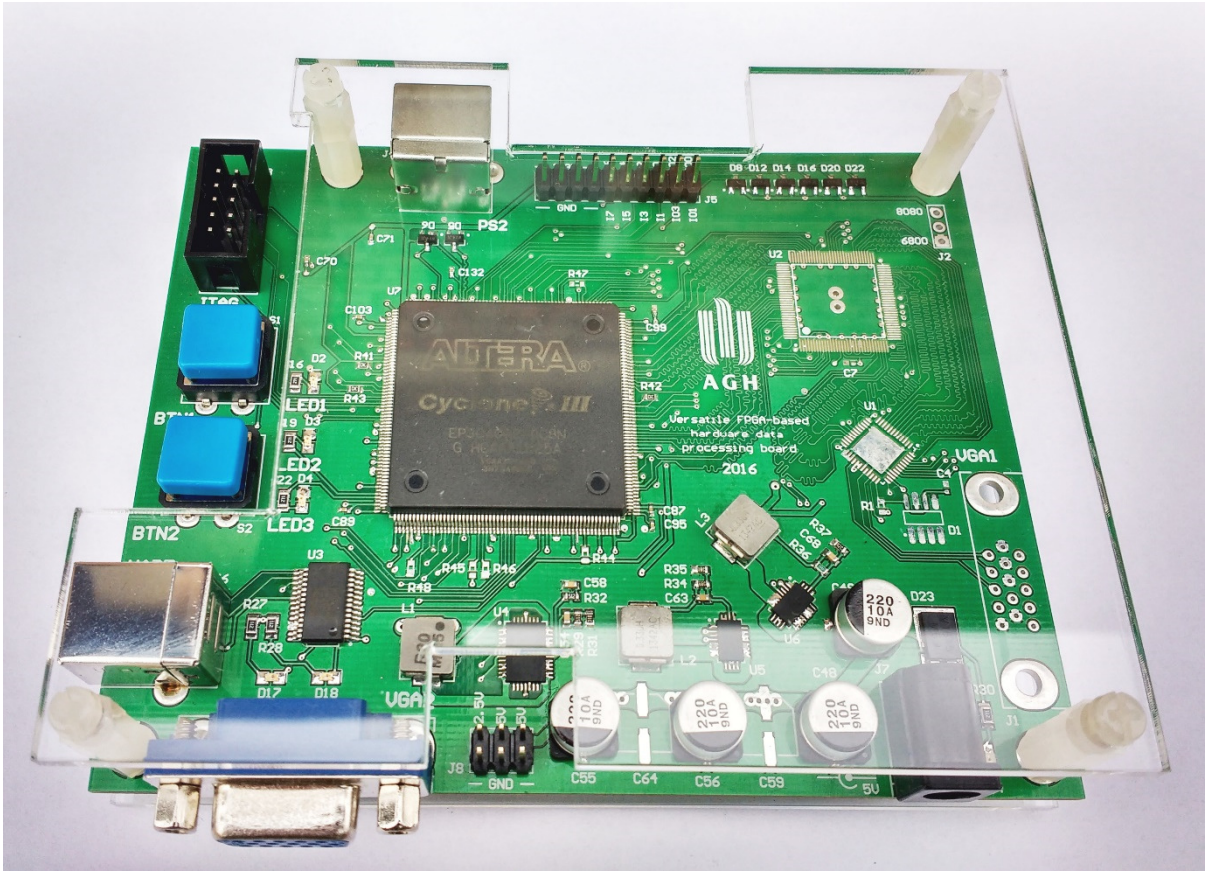
Projekt obwodu drukowanego modułu SNF-0 został wykonany w module *PCB Editor* programu *Altium Designer*. W początkowej fazie prac przygotowano prototyp, który pozwolił na oszacowanie możliwości i wymagań odnośnie struktury i aspektów technicznych obwodu drukowanego. Więcej w tym temacie napisano w dokumentacji procesowej.

Porty wejścia/wyjścia, przyciski i kontrolki zostały ułożone na brzegach płytki, umożliwiając łatwy dostęp. W centralnej części PCB znajduje się układ FPGA, po prawej stronie rozbudowany moduł wyświetlania VGA1, natomiast u dołu umieszczono układ zasilania.

Wykorzystano technologię *length tuning* dla dopasowania długości magistrali komunikacyjnych między układem FPGA i pamięcią oraz FPGA i buforem obrazu. Połączenia między układem SSD6319 i przetwornikiem cyfrowo-analogowym także zostały dopasowane pod względem długości. Pozwoli to na wyrównanie czasów opóźnień w przesyle danych i uniknięcie związanych z tym problemów przy wyko-

rzystaniu wysokich częstotliwości. Starano się także, aby ważne połączenia były jak najkrótsze. Odpowiednio zwiększono grubość ścieżek układu zasilania, którymi może płynąć prąd o wysokim natężeniu. Zastosowano opisy na obu stronach płytki PCB, które ułatwiły montaż elementów i stanowią podstawowy opis dostępnych portów i kontrolek. W miejscach, w których nie ma ścieżek i elementów zastosowano wypełnienie obszarem miedzi podłączonym do masy, w celu redukcji szumów i ułatwienia dostępu masy do wymaganych miejsc.

3. 5. Wykonanie płytki obwodu drukowanego i montaż elementów



Rys. 6. Gotowy moduł SNF-0 (widok od góry)

Wykonanie płytki obwodu drukowanego zostało zlecone firmie zajmującej się tego typu usługami. Po skompletowaniu wszystkich elementów i odebraniu płytki, przystąpiono do jej montażu. Zaprojektowano i zamówiono także prostą obudowę ze szkła akrylowego, zabezpieczającą elementy przed przepięciami elektrostatycznymi i urazami mechanicznymi, która składa się z dwóch części montowanych za pomocą plastikowych wsporników i śrubek.

Montaż elementów wykonano przy użyciu stacji lutowniczej wyposażonej w standardową lutownicę grotową i dmuchawę na gorące powietrze. Zastosowanie urządzenia *hot air* między innymi ułatwiło montaż niektórych małych elementów (np. generatora kwarcowego).

Obwód drukowany ma wymiary 9,5 x 12 cm i został wykonany z laminatu o grubości 1.5 mm, pokrytego 35 mikrometrową warstwą miedzi. Zastosowano cynowanie metodą HASL (*hot air solder laveling*).

3. 6. Uruchomienie i testowanie modułu SNF-0

Testowanie modułu SNF-0 odbywało się bezpośrednio po montażu każdego podsystemu. Najpierw przygotowano układ zasilania i upewniono się, że napięcia przez niego wytwarzane są odpowiednio stabilne. Przetestowano spadki napięcia pod obciążeniem.

Następnie przylutowano generator kwarcowy i sprawdzono czy sygnał zegarowy na jego wyjściu spełnia wymagania. Kolejnym etapem było zmontowanie interfejsu USB do portu szeregowego i przetestowanie komunikacji z komputerem. Po przylutowaniu innych wymaganych elementów dyskretnych, przystąpiono do montażu układu FPGA.

Po zmontowaniu całego modułu, sprawdzono możliwość programowania układu Cyclone III i przygotowano prostą konfigurację testową. Następnie, po ustawieniu odpowiedniego stanu wyjść, sprawdzono oscyloskopem poziomy napięć na nich.

Po przejściu wstępnych testów, przygotowano konfiguracje testujące podstawowy port VGA2, działanie przycisków, diod, złącza PS2 i portu szeregowego.

Następnie przystąpiono do uruchamiania docelowego projektu specjalizowanego procesora graficznego wyświetlającego prostą grafikę 3D zbudowaną z linii. Jego opis znajduje się w kolejnym rozdziale.

3. 7. Projekt i wykonanie przykładowego specjalizowanego procesora graficznego

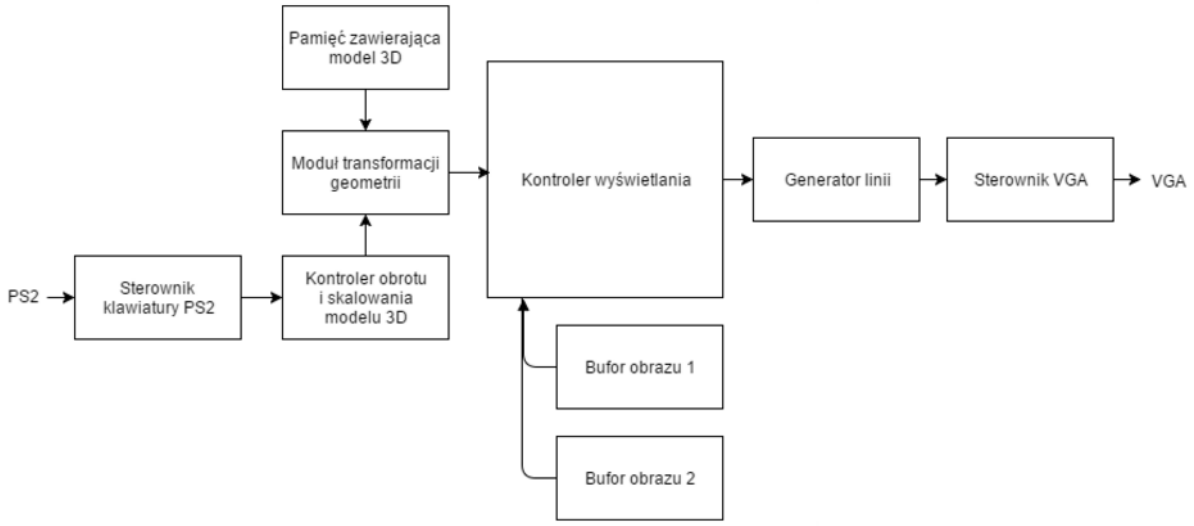
Głównym celem projektu inżynierskiego było zbudowanie platformy umożliwiającej tworzenie procesorów zaprojektowanych w językach opisu sprzętu. Aby zaprezentować możliwości modułu SNF-0 wykonano przykładowy specjalizowany procesor graficzny służący do wyświetlania prostej grafiki 3D zbudowanej z linii. Został on napisany w języku VHDL.

Projekt wykorzystuje wyjście VGA2 do wyświetlania jednokolorowego obrazu. Ograniczenie rozdzielczości i ilości barw związane jest z niewielką pamięcią wewnętrzną układu FPGA, która częściowo jest wykorzystywana przez projekt, jako bufory obrazu (zastosowano podwójne buforowanie).

Do kontroli przekształceń rysowanej bryły wykorzystywana jest klawiatura podłączona do portu PS2. Użytkownik może sterować obrotem oraz skalowaniem obiektu 3D.

Procesor graficzny posiada konfigurację w postaci geometrii bryły zapisaną na stałe w jego kodzie, która może jednak być w łatwy sposób podmieniana poprzez dostarczenie danych w odpowiednim formacie i ponowną kompilację projektu.

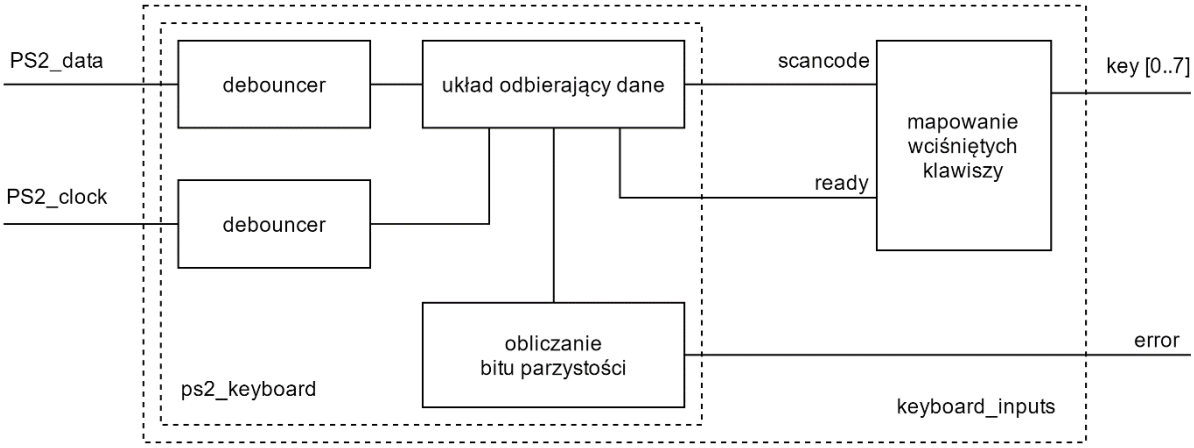
Napisano skrypt eksportu modelu 3D z programu 3DS Max, który generuje plik **.vhd*, zawierający odpowiednią konfigurację procesora graficznego.



Rys. 7. Schemat blokowy specjalizowanego procesora graficznego

Poniżej przedstawiono opis poszczególnych jednostek procesora graficznego.

3. 7. 2. Kontroler klawiatury PS2



Rys. 8. Schemat blokowy kontrolera klawiatury PS2

PS2 jest interfejsem pozwalającym na podłączenie klawiatury lub myszki do komputera, posiadającego 6-pinowe złącze Mini-DIN. Urządzenie wyposażone w gniazdo PS2 musi dostarczyć klawiaturze lub myszce napięcie zasilające o wartości 5V. Komunikacja przebiega przy użyciu 2 sygnałów: zegarowego i linii danych. Sygnał zegarowy portu PS2 ma częstotliwość od 10 do 16.7 KHz. Przesył danych poprzedzany jest bitem startowym (stan niski). Następnie transmitowany jest jeden bajt danych, bit parzystości i bit końcowy. Po skończeniu przesyłania oba sygnały wracają do stanu wysokiego.
Odebrane dane reprezentują część tzw. *scan code*. Kod naciśnięcia klawisza (*make code)* zazwyczaj składa się z jednego lub dwóch bajtów. *Break code* sygnalizuje zwolnienie przycisku klawiatury, który oprócz informacji o kodzie klawisza, zawiera dodatkowy bajt o wartości *F0*.

Kontroler klawiatury składa się z dwóch jednostek: *ps2_keyboard* i *keyboard_inputs*. Do sterownika podłączone są sygnały portu PS2, które synchronizowane są przy użyciu rejestrów i filtrowane przez jednostki projektowe *debouncer*. Odebrane dane zapisywane są do rejestru przesuwnego, taktowanego zegarem *PS2_clock*. Układ odbierający dane posiada także licznik, który odmierza ilość cykli do zakończenia pobierania bajtu. Kiedy transmisja zostanie wykonana, ustawiana jest flaga *ready*. Do sprawdzenia poprawności odebranych danych wykorzystywany jest bit parzystości.

Druga część jednostki kontrolera klawiatury *keyboard_inputs* odpowiedzialna jest za zmapowanie odebranych bajtów na flagi określające wciśnięcie klawisza. Każdy bit wyjścia *key* zawiera informację, czy wciśnięto klawisz odpowiadający jego indeksowi.

3. 7. 3. Moduł kontroli skalowania i obrotu

Jednostka modułu kontroli skalowania i obrotu (*input_handler*) wykorzystywana jest do kontroli przekształceń wyświetlanej bryły 3D. Na podstawie tablicy wciśniętych klawiszy, przekazanej z modułu *keyboard_inputs*, zmieniany jest kąt obrotu wokół każdej z osi oraz skalowanie wyświetlanego obiektu.

Moduł posiada licznik, inkrementowany, gdy ramka obrazu VGA zaczyna być rysowana, który powoduje opóźnienie aktualizacji przekształceń modelu 3D. Wartości kątów obrotu i skali przekazywane są do jednostki transformacji geometrycznych.

3. 7. 4. Moduł pamięci i struktura danych modelu 3D

Do przechowywania danych geometrycznych trójwymiarowego modelu, który ma zostać wyświetlony na specjalizowanym procesorze graficznym, wykorzystano bloki pamięci układu FPGA, skonfigurowane do pracy w trybie ROM. Jednostka projektowa *model_memory* wykorzystywana jest do synchronicznego odczytu danych. W pakiecie *model* znajduje się wyeksportowana struktura bryły 3D.

Geometria, przechowywana w blokach ROM, reprezentowana jest przez tablicę 96-bitowych wektorów. Każdy element składa się z trójwymiarowych współrzędnych końców krawędzi modelu. Składowe współrzędnych reprezentowane są przez 16-bitowe liczby całkowite.

Wszystkie dalsze obliczenia dostosowane są do liczb z przedziału -8191 do 8191. Jest to wystarczająca dokładność, aby uzyskać zadowalające efekty przy dobrym wykorzystaniu dostępnych zasobów pamięciowych i logicznych. Wydobycie składowych z elementu tablicy ma miejsce w module przekształceń geometrycznych.

Każdy model 3D, przed użyciem w opisywanym projekcie, musi zostać wyeksportowany z programu *Autodesk 3DS Max* za pomocą przygotowanego skryptu. Eksporter ten tworzy plik **.vhd*, który zawiera dane modelu. Więcej na ten temat w podrozdziale 5. 11.

3. 7. 5. Moduł przekształceń geometrycznych

Wstęp

Jednostka projektowa wykonująca transformacje geometryczne jest najważniejszym modułem projektu. Jej funkcją jest obliczanie przekształceń obrotu i skalowania modelu 3D, a także rzutowanie trójwymiarowych współrzędnych na płaszczyznę wyświetlaną na monitorze przy pomocy złącza VGA.

Na wejściu jednostki podawane są kąty obrotu wokół każdej z osi oraz współczynnik skalowania bryły, a także sygnały sterujące z modułu kontrolera wyświetlania. Jednostka przekształceń geometrycznych posiada instancję pamięci modelu, przechowującej dane określające współrzędne krawędzi tworzących model 3D. Na wyjściu pojawiają się współrzędne *x0, y0, x1, y1* kolejnych przekształconych krawędzi. Cały proces jest synchronizowany przez kontroler wyświetlania.

Jednostka dostarczająca wartości funkcji trygonometrycznych

Do obliczeń trygonometrycznych potrzebne są wartości funkcji sinus i cosinus kątów obrotu bryły. Dostarczane są one przez jednostkę *sin_cos*. Podstawowym elementem tego modułu jest tablica, zawierająca wartości funkcji sinus dla całkowitych kątów od 0° do 90°. Dokładność ta okazała się dostarczać zadowalających efektów wizualnych obrotu bryły.

Poniżej przedstawiono sposób wyznaczania wartości funkcji trygonometrycznych sinus i cosinus dla kątów w zakresie 0° do 360°. Tablica *sin_lut* zawiera 90 obliczonych wcześniej wartości. Do zmiennych wyjściowych *sin_out* i *cos_out* przypisywana jest wartość funkcji dla danej wejściowej *angle*.

```
if angle ≥ 270° then
    sin_out ← -sin_lut[360° - angle]
    cos_out ←  sin_lut[angle - 270°]
elsif angle ≥ 180 then
    sin_out ← -sin_lut[angle - 180°]
    cos_out ← -sin_lut[270° - angle]
elsif angle ≥ 90° then
    sin_out ←  sin_lut[180° - angle]
    cos_out ← -sin_lut[angle - 90°]
else
    sin_out ←  sin_lut[angle]
    cos_out ←  sin_lut[90° - angle]
end if
```

Działanie modułu transformacji

Podstawowym elementem jednostki transformacji jest rozbudowana maszyna stanów, która steruje wykonaniem całego przekształcenia. W początkowej fazie przygotowywane są wartości funkcji trygonometrycznych dla poszczególnych kątów. Następnym krokiem jest dokonanie obliczeń potrzebnych do wykonania obrotu i skalowania. Końcowym etapem jest oczekiwanie, aż linia, o końcach w przekształconych współrzędnych, zostanie narysowana. Powyższe operacje powtarzane są, aż cała scena zostanie przygotowana do wyświetlenia. Szczegółowy opis kolejnych kroków maszyny stanów opisano w poniższej tabeli.

Nazwa stanu	Następny stan	Opis
request_sincos_x	*get_sincos_x*	Przekazanie kąta obrotu wokół osi x do modułu *sin_cos*
get_sincos_x	*request_sincos_y*	Zapisanie wartość funkcji sinus i cosinus dla wcześniej podanego kąta
request_sincos_y	*get_sincos_y*	Przekazanie kąta obrotu wokół osi y do modułu *sin_cos*
get_sincos_y	*request_sincos_z*	Zapisanie wartość funkcji sinus i cosinus dla wcześniej podanego kąta
request_sincos_z	*get_sincos_z*	Przekazanie kąta obrotu wokół osi z do modułu *sin_cos*
get_sincos_z	*cast_edge_to_32bit*	Zapisanie wartość funkcji sinus i cosinus dla wcześniej podanego kąta
cast_edge_to_32bit	*apply_x_rot*	Skopiowanie danych aktualnej krawędzi do tymczasowej struktury do obliczeń o 32-bitowych wartościach
apply_x_rot	*apply_y_rot*	Wykonanie obliczeń skutkujących obrotem obiektu wokół osi x
apply_y_rot	*apply_z_rot*	Wykonanie obliczeń skutkujących obrotem obiektu wokół osi y
apply_z_rot	*apply_scale*	Wykonanie obliczeń skutkujących obrotem obiektu wokół osi z
apply_scale	*inc_line_cnt*	Przeskalowanie współrzędnych krawędzi przez podany współczynnik

inc_line_cnt	*edge_draw_wait* – jeżeli pozostały krawędzie do wyświetlenia *next_wait* – jeżeli wszystkie krawędzie są narysowane	Zwiększenie licznika krawędzi i ustawienie flagi *edge_ready* – gdy pozostały jeszcze krawędzie do przekształcenia, wyzerowanie licznika i ustawienie bitu wyjściowego *rendered* w przeciwnym wypadku
edge_draw_wait	*cast_edge_to_32bit*	Oczekiwanie na pojawienie się zlecenia przekształcenia kolejnej krawędzi od kontrolera wyświetlania (po skończeniu jej rysowania). Obliczenia wartości funkcji sinus i cosinus dla wymaganych kątów nie zmieniają się, transformowana jest tylko krawędź, wskazywana przez licznik *inc_line_cnt*.
next_wait	*request_sincos_x* – jeżeli ustawiono flagę *update_rot_request* *cast_edge_to_32bit* – w przeciwnym wypadku	Oczekiwanie na sygnał rozpoczęcia kolejnej transformacji krawędzi modelu 3D. Jeżeli ustawiona jest flaga świadcząca o zmianie któregoś z kątów obrotu (*update_rot_request*), aktualizowane są wartości funkcji trygonometrycznych. W przeciwnym wypadku, rozpoczyna się przekształcanie kolejnych krawędzi.

Oprócz logiki sekwencyjnej implementującej wyżej opisaną maszynę stanów, zastosowano układy kombinacyjne.

Pierwszy z nich jest odpowiedzialny za wyodrębnienie poszczególnych współrzędnych obu końców krawędzi z 96-bitowej wartości przechowywanej w pamięci modelu. Zostaje ona podzielona na 12 szesnastobitowych liczb.

Drugi układ konwertuje przekształcone współrzędne do przestrzeni 2D i dokonuje odpowiedniego skalowania według poniższego algorytmu. Wartości *x0, y0, x1, y1* (w przedziale [-8191, 8191]) oznaczają współrzędne końców linii do narysowania, natomiast struktura *edge* zawiera koordynaty przekształconej krawędzi w 3D.

```
x0 ← edge.z0 / 64 + 640 / 2
y0 ← edge.y0 / 64 + 480 / 2
x1 ← edge.z1 / 64 + 640 / 2
y1 ← edge.y1 / 64 + 480 / 2
```

Przekształcenia geometryczne

Omawiany procesor graficzny wykonuje dwa rodzaje przekształceń bryły 3D: rotacje wokół osi x, y i z oraz skalowanie.

Dla uproszczenia projektu i optymalizacji liczby wymaganych zasobów układu FPGA, wszystkie transformacje wykonywane są na szesnastobitowych liczbach całkowitych, wykorzystując operacje mnożenia i dodawania. Na każdym etapie wykonywania przekształceń odrzucane są najmniej znaczące bity wyniku poprzez operację przesunięcia arytmetycznego w prawo. Początkowa bryła posiada wierzchołki odpowiednio przeskalowane na etapie eksportu tak, aby znajdowały się w przedziale -8191 do 8191. Dobrano wartości przesunięć na każdym etapie obliczeń w taki sposób, aby końcowy wynik również był z takiego przedziału.

Podstawą wykorzystanego algorytmu skalowania i obrotu były operacje macierzowe (przedstawione na rysunku 9). Wektor współrzędnych wierzchołka jest mnożony przez kolejne macierze transformacji. Przekształcono działania tak, aby wystarczyło obliczenie sumy iloczynów. Rozdzielono obliczanie poszczególnych transformacji w celu zmniejszenia opóźnień w układzie logicznym.

(a)

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(b)

$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix}$$

(c)

$$V_{out} = R_x R_y R_z S \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Rys. 9. (a) Macierze obrotu dla poszczególnych współrzędnych, (b) macierz skalowania, (c) złożenie przekształceń i obliczenie wektora wynikowego

3. 7. 6. Kontroler wyświetlania

Jednostka kontrolera wyświetlania jest kolejnym ważnym modułem specjalizowanego procesora graficznego. Jej głównym zadaniem jest zarządzanie przebiegiem transformacji wejściowych krawędzi, rysowania linii i zapisem do odpowiedniego bufora obrazu.

Zastosowano technikę podwójnego buforowania polegającą na tym, że scena renderowana jest do jednego z nich, podczas gdy z drugiego pobierane są kolejne piksele do wyświetlenia na ekranie. Po skończeniu procesu, bufory zostają zamienione. Dzięki zastosowaniu tej metody, uniknięto efektu migotania rysowanej sceny i zwiększono czas przeznaczony na wygenerowanie obrazu.

Jako pamięć obrazu, wykorzystano układy 2-PORT RAM z biblioteki *Megafunctions,* dostępnej w oprogramowaniu *Altera Quartus II*. Dzięki możliwości użycia osobnego zegara do odczytu i zapisu, możliwe było zsynchronizowanie wyświetlania obrazu przez złącze VGA, przy jednoczesnym zachowaniu wyższego taktowania generowania sceny (zapis: 50MHz, odczyt: 25MHz).

Przechowywany obraz ma 1-bitową głębię koloru. Zastosowanie 2 buforów zajęło ok. 50% jednostek pamięci M9K dostępnych w układzie FPGA. Do przełączania buforów stworzono odpowiednią logikę zsynchronizowaną ze sterownikiem VGA. Przełączania pamięci do zapisu i wyświetlania następuje po skończeniu wyświetlania obrazu.

Proces odczytu danych obrazu oblicza adres pamięci RAM, na podstawie współrzędnych aktualnie wyświetlanego piksela, dostarczonych przez moduł kontrolera VGA. Dane koloru piksela pobierane są z bufora i przekazywane do kontrolera VGA. Po skończeniu przesyłania danych koloru, następuje zamiana buforów.

Proces zapisujący do pamięci RAM jest nieco bardziej skomplikowany. Zbudowany został, jako maszyna stanów. Jej opis znajduje się w poniższej tabeli.

Nazwa stanu	Następny stan	Opis
start_draw	*clear*	Aktualny adres bufora zapisu ustawiany jest na 0, w stan wysoki przechodzi flaga *start*, która informuje o rozpoczęciu generowania sceny jednostce transformującej geometrię, która przygotowuje pierwszą krawędź.
Clear	*start_trasform* – po skończeniu czyszczenia bufora	Następuje przypisanie koloru tła do wszystkich pikseli bufora zapisu.
start_trasform	*transform_wait* – jeżeli nie przekształcono jeszcze wszystkich krawędzi *frame_rendered* – w przeciwnym przypadku	Ustawienie flagi zlecającej rozpoczęcie transformacji kolejnej krawędzi, jeżeli jeszcze nie wszystkie zostały przekształcone.
transform_wait	*load* – jeżeli scena nie została wyrenderowana i krawędź dostępna jest przekształcona krawędź *frame_rendered* – jeżeli scena została wyrenderowana	Oczekiwanie na przekształcenie krawędzi przez moduł *transformer*.

Load	*init_draw*	Skopiowanie przekształconych współrzędnych do rejestrów generatora linii.
init_draw	*draw_edge*	Rozpoczęcie rysowania linii
draw_edge	*start_transform* – jeżeli cała linia została wygenerowana	Pobieranie kolejnych współrzędnych pikseli z generatora linii i ich zapis w pamięci obrazu.
frame_rendered	*start_draw* – po zamianie buforów	Oczekiwanie na zlecenie zamiany buforów od procesu wyświetlającego obraz na monitorze VGA i wykonanie tej akcji.

3. 7. 7. Generator linii

Do generowania linii zastosowano algorytm Bresenhama. Po zleceniu rysowania linii, moduł generuje współrzędne kolejnych pikseli i ustawia flagę rysowania. Do synchronizacji zastosowano prostą maszynę stanów. Sterowanie jednostką generatora odbywa się z poziomu kontrolera wyświetlania. Obliczanie współrzędnych pikseli zrealizowano przy pomocy logiki kombinacyjnej.

3. 7. 8. Kontroler VGA

Jednostka projektowa kontrolera VGA na wyjściu, oprócz wartości kolorów, generuje sygnały synchronizacji poziomej i pionowej. Dostępne są także rejestry *x_pos* i *y_pos*, które zawierają współrzędne aktualnie wyświetlanego piksela. Jeżeli generowany obraz jest aktualnie widoczny flaga *visible* jest ustawiana. Na wejściu kontrolera podawany jest sygnał zegarowy oraz wartość koloru piksela.

Najważniejszym elementem jednostki projektowej *vga* są generatory sygnałów *h_sync* i *v_sync* sterowane licznikami taktowanymi zegarem *clk* o częstotliwości 25.175 MHz.

3. 7. 9. Eksport pliku z programu 3DS Max do pamięci układu FPGA

Program eksportujący model 3D z programu *Autodesk 3DS Max* został napisany w języku skryptowym tego środowiska (*MaxScript*). Język ten pozwala na łatwy dostęp do elementów sceny wczytanych do oprogramowania firmy *Autodesk*, a także umożliwia operację na plikach. Wykorzystanie *MaxScript* pozwoliło w łatwy sposób przygotować model 3D do zapisania w pamięci układu FPGA.

Po załadowaniu skryptu eksportu, pojawia się przycisk z napisem *Export scene objects*, po którego naciśnięciu wykonywany jest zapis sceny do pliku **.vhd*.

Po uruchomieniu, skrypt wyświetla dialog wyboru ścieżki do zapisu. Następnie wykonywane są następujące operacje dla obiektów sceny: przeskalowanie bryły tak, aby współrzędne jej wierzchołków mieściły się w przedziale od -1 do 1, konwersja do trybu *Editable Poly* i zapis danych do pliku.

W pliku **.vhd* tworzona jest struktura pakietu (*package*) języka VHDL, w której znajdują się dwie stałe: przechowująca ilość krawędzi oraz tablica z danymi. Podczas zapisu, skrypt eksportera pobiera współrzędne wszystkich krawędzi obiektów sceny, skaluje je przez 8191 i tworzy z nich łańcuch znaków złożony z 12 liczb szesnastobitowych. Wartości zapisane są w postaci szesnastkowej.

Tak przygotowany plik **.vhd* może być w łatwy sposób użyty w projekcie specjalizowanego procesora graficznego.

4. Organizacja pracy

4. 1. Opis zastosowanej metodyki

Do realizacji projektu przyjęto metodykę przyrostową. Po realizacji każdego przyrostu odbyło się spotkanie z Promotorem. Prace podzielono na 4 iteracje.

4. 2. Pierwszy przyrost

Cele

- Ogólna koncepcja modułu
- Wybór układu FPGA
- Dobór elementów peryferyjnych
- Wstępny schemat projektu
- Rozpoznanie dostępności elementów w internetowych hurtowniach elektronicznych

Wyniki i podsumowanie

W końcowej fazie tego etapu gotowy był wstępny schemat modułu oraz lista potrzebnych elementów. Wybrane komponenty w większości trafiły do końcowego produktu. Wyjątkiem są układy stabilizatorów LDO, które mogły nie spełniać założonych parametrów. Założenia tego etapu projektu zostały spełnione.

4. 3. Drugi przyrost

Cele

- Ostateczna wersja schematu modułu SNF-0
- Prototyp obwodu drukowanego
- Ostateczna lista elementów
- Zakup potrzebnych elementów elektronicznych

Wyniki i podsumowanie

Na koniec 2 iteracji gotowy był prototyp obwodu drukowanego modułu SNF-0. Dzięki jego wykonaniu zapoznano się z oprogramowaniem *Altium Designer* oraz zidentyfikowano cele i wymagania przed tworzeniem ostatecznej wersji płytki PCB. Na tym etapie gotowa była także lista potrzebnych elementów.

4. 4. Trzeci przyrost

Cele

- Ostateczna wersja obwodu drukowanego
- Zamówienie wykonania płytki PCB
- Montaż elementów
- Uruchomienie zestawu
- Zaprojektowanie i przygotowanie obudowy

Wyniki i podsumowanie

W trzeciej iteracji zakończono prace nad sprzętową częścią modułu SNF-0. Cele przyrostu zostały osiągnięte. Elementy potrzebne do wykonania projektu inżynierskiego zostały zmontowane. Układy takie jak pamięć SSRAM oraz pamięć obrazu zostaną wykorzystane do przyszłych projektów. Moduł SNF-0 w obecnym stanie wyposażony jest w działający i zdolny do zaprogramowania układ FPGA oraz zestaw portów wejścia/wyjścia (PS2, UART, GPIO, VGA).

4. 5. Czwarty przyrost

Cele

- Napisanie i uruchomienie modułów VHDL odpowiedzialnych za obsługę portów PS2, szeregowego i wyjścia VGA
- Implementacja i uruchomienie przykładowego procesora graficznego wyświetlającego grafikę 3D zbudowaną z linii
- Finalizacja projektu i ostateczne poprawki

Wyniki i podsumowanie

Po skończeniu ostatniego przyrostu, projekt uniwersalny modułu sprzętowego przetwarzania danych opartego na FPGA był gotowy. Zakończone zostały prace nad częścią programową, których wynikiem był przykładowy specjalizowany procesor graficzny prezentujący możliwości zestawu SNF-0.

5. Analiza wykonanych prac

5. 2. Analiza realizacji założeń projektowych

Celem projektu inżynierskiego było przygotowanie uniwersalnej platformy umożliwiającej między innymi tworzenie własnych implementacji procesorów. Zastosowanie układu FPGA *Altera Cyclone III*, posiadającego stosunkowo duże zasoby, pozwala na tworzenie rozbudowanych projektów.

Przygotowanie podstawowego zestawu portów wejścia/wyjścia takich, jak: PS2, UART, wyjście VGA spełnia wymagania postawione przed projektem.

Zaimplementowany specjalizowany procesor graficzny prezentuje możliwości zestawu, przede wszystkim pokazuje możliwość szybkiego przetwarzania stosunkowo dużej ilości danych i generowania obrazów. Prezentuje także praktyczne wykorzystanie dostępnych portów wejścia/wyjścia.

5. 3. Plany rozwoju projektu

Moduł uruchomieniowy SNF-0 posiada kilka podsystemów, które nie zostały jeszcze zmontowane. Pamięć SSRAM pozwoli na przechowywanie większej porcji danych i umożliwi wykorzystanie potencjału układu FPGA. Wyjście wideo VGA1 z buforem obrazu i 24-bitowym przetwornikiem cyfrowo-analogowym pozwoli na wyświetlanie skomplikowanej grafiki w pełnej palecie kolorów.

Zestaw SNF-0 jest platformą, na której można tworzyć dowolne projekty, ograniczone tylko zasobami układu FPGA i dostępnymi peryferiami.

Specjalizowany procesor graficzny posiada także duży potencjał rozwoju. W planach jest wykorzystanie wielu jednostek generujących obraz do przyspieszenia wyświetlania. Dzięki pamięci SSRAM, możliwe będzie przechowywanie bardziej złożonych modeli. Montaż podsystemu wyświetlania VGA1 pozwoli na rozpoczęcie prac nad generowaniem brył wypełnionych kolorem, a nawet zastosowanie cieniowania i oświetlenia.

6. Materiały źródłowe

[1] Altera Corporation. *Cyclone III Device Handbook Volume 1* https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyc3/cyclone3_handbook.pdf (dostęp: 3.12.2016)

[2] Altera Corporation. *Cyclone III Device Handbook Volume 2* https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyc3/cyc3_ciii5v2.pdf (dostęp: 3.12.2016)

[3] FTDI Chips. *FT232R USB UART IC Datasheet, 2015* http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf (dostęp: 3.12.2016)

[4] Solomon Systech. *SSD1963 Product Review*, 2008 http://www.allshore.com/pdf/solomon_systech_ssd1963.pdf (dostęp: 5.10.2016)

[5] ISSI. Datasheet: *IS61vPS51236A*, 2015 http://www.issi.com/WW/pdf/61VPS_LPS-51236A_102418A.pdf (dostęp: 5.10.2016)

[6] Analog Devices. *ADV7125 Datasheet,* 2002-2016 http://www.analog.com/media/en/technical-documentation/data-sheets/ADV7125.pdf (dostęp: 3.12.2016)

[7] Pong P. Chu. *FPGA Prototyping by VHDL Examples*. John Wiley & Sons, Inc., 2008

[8] Włodzimierz Wrona. *VHDL - język opisu i projektowania układów cyfrowych.* Pracownia Komputerowa Jacka Skalmierskiego, 1998

[9] OpenGL Transformatios https://open.gl/transformations (dostęp: 10.11.2016)

7. Spis rysunków