

**AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE**

Wydział Informatyki, Elektroniki i Telekomunikacji
Katedra Informatyki



PROJEKT INŻYNIERSKI
Uniwersalny moduł sprzętowego
przetwarzania danych oparty na FPGA

Autor: Krzysztof Papciak
Opiekun: dr inż. Jacek Długopolski

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. Dziedzina problemu	5
1. 1. Układy FPGA	5
1. 2. Rodzina układów FPGA <i>Cyclone III</i> firmy <i>Altera</i>	6
1. 2. 1. Elementy i bloki logiczne w rodzinie układów <i>Cyclone III</i>	6
1. 2. 2. Bloki pamięci w rodzinie układów <i>Cyclone III</i>	8
1. 2. 3. 18-bitowe jednostki mnożące w układzie FPGA <i>Cyclone III</i>	8
1. 2. 4. Pętle sprzężenia fazowego w układach rodziny <i>Cyclone III</i>	9
1. 3. Zastosowanie układu FPGA <i>Altera Cyclone III</i> w projekcie	9
2. Studium wykonalności	10
2. 1. Określenie celów	10
2. 2. Analiza ryzyka	10
3. Metodyka pracy.....	13
3. 1. Opis zastosowanej metodyki	13
3. 2. Narzędzia wykorzystywane w procesie tworzenia modułu i projektu procesora	13
3. 3. Przebieg prac	14
3. 3. 1. Pierwszy przyrost.....	14
3. 3. 2. Drugi przyrost	16
3. 3. 3. Trzeci przyrost	18
3. 3. 4. Czwarty przyrost.....	22
4. Opis szczegółów implementacyjnych.....	24
4. 1. Schemat modułu SNF-0	24
4. 1. 1. Układ FPGA i jego otoczenie.....	25
4. 1. 2. Pamięć SSRAM.....	27
4. 1. 3. Port szeregowy	28
4. 1. 4. Port PS2	29
4. 1. 5. Przyciski i diody LED	29
4. 1. 6. Porty wejścia/wyjścia ogólnego przeznaczenia i wyprowadzenia napięć zasilających.....	30
4. 1. 7. Wyjście VGA2	31
4. 1. 8. Wyjście VGA1, przetwornik cyfrowo analogowy i bufor obrazu.....	31
4. 1. 9. Moduł zasilający	34
4. 2. Projekt i wykonanie obwodu drukowanego modułu SNF-0	36
4. 2. 1. Projekt obwodu drukowanego.....	36
4. 2. 2. Wykonanie płytka obwodu drukowanego i montaż elementów	36

4. 2. 3. Uruchomienie i testowanie modułu SNF-0	37
4. 3. Projekt i wykonanie przykładowego procesora graficznego	38
4. 3. 1. Kontroler klawiatury PS2	39
4. 3. 2. Moduł kontroli skalowania i obrotu	39
4. 3. 3. Moduł pamięci i struktura danych modelu 3D.....	40
4. 3. 4. Moduł przekształceń geometrycznych.....	40
4. 3. 5. Kontroler wyświetlania.....	44
4. 3. 6. Generator odcinków reprezentujących krawędzie obiektu 3D.....	46
4. 3. 7. Kontroler VGA.....	46
4. 3. 8. Eksport pliku z programu 3DS Max do pamięci układu FPGA	47
5. Prezentacja gotowego produktu	48
5. 1. Opis elementów płytki PCB modułu SNF-0.....	48
5. 2. Szczegółowe parametry podzespołów modułu SNF-0	51
5. 3. Instrukcja uruchomienia i wykorzystania modułu SNF-0	53
5. 3. 1. Peryferia sprzętowe	53
5. 3. 2. Zalecane oprogramowanie.....	54
5. 3. 3. Wspierane wersje oprogramowania Quartus II	54
5. 4. Opis wykorzystania procesora graficznego	55
5. 4. 1. Kompilacja i uruchomienie projektu na module SNF-0.....	55
5. 4. 2. Sterowanie przy pomocy klawiatury PS2	56
5. 5. Instrukcja przygotowania własnych modeli do wyświetlenia na procesorze graficznym	57
5. 5. 1. Przygotowanie modelu w programie Autodesk 3DS Max.....	57
5. 5. 2. Eksport modelu przy użyciu skryptu.....	58
5. 5. 3. Przygotowanie do wyświetlenia modelu na module SNF-0	59
6. Analiza wykonanych prac.....	61
6. 1. Analiza realizacji założeń projektowych	61
6. 2. Plany rozwoju projektu.....	61
7. Materiały źródłowe	62
8. Spis rysunków	63

1. Dziedzina problemu

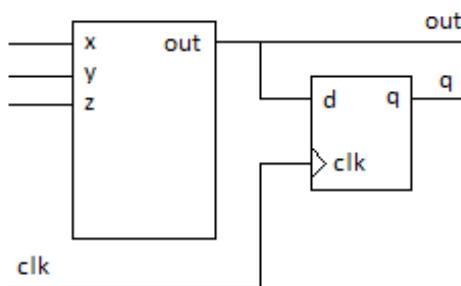
Celem projektu inżynierskiego było stworzenie uniwersalnej platformy do sprzętowego przetwarzania danych, zrealizowanej w oparciu o układ FPGA. Moduł docelowo ma służyć do łatwego projektowania, uruchamiania i testowania procesorów zaprojektowanych w językach opisu sprzętu, takich jak VHDL i Verilog.

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.

1. 1. Układy FPGA

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.



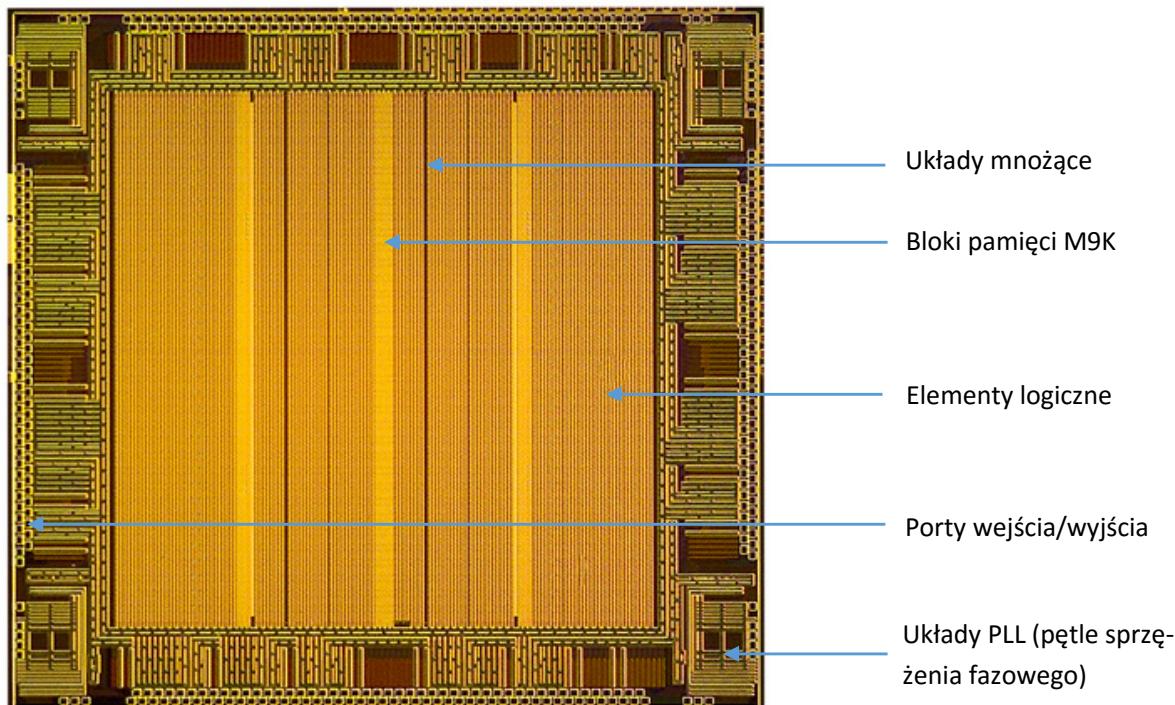
Rys. 1. Schemat koncepcyjny jednostki logicznej opartej o tablicę LUT

Podstawowym elementem budującym układ FPGA jest element logiczny (LE), który zawiera mały konfigurowalny obwód kombinacyjny z przerutnikiem 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 wejścia przyporządkowany jest odpowiedni stan wyjścia [8]. Schemat koncepcyjny jednostki logicznej opartej na LUT przedstawiony został na rysunku 1.

1. 2. Rodzina układów FPGA *Cyclone III* firmy Altera

Do wykonania projektu inżynierskiego wybrano układ FPGA *Altera Cyclone III*.

Rodzina układów *Cyclone III* charakteryzuje się między innymi niskim poborem mocy. Układy tej rodziny posiadają od ok. 5 tys. do 200 tys. elementów logicznych oraz od 0.5Mb do 8Mb pamięci RAM [2].



Rys. 2. Struktura układu FPGA rodziny Cyclone II i opis podstawowych elementów

Na rysunku 2 przedstawiono zdjęcie przedstawiające strukturę układu FPGA Altera Cyclone III. Widoczne są między innymi bloki układów mnożących, elementy pamięci RAM oraz struktury wejścia/wyjścia.

1. 2. 1. Elementy i bloki logiczne w rodzinie układów *Cyclone III*

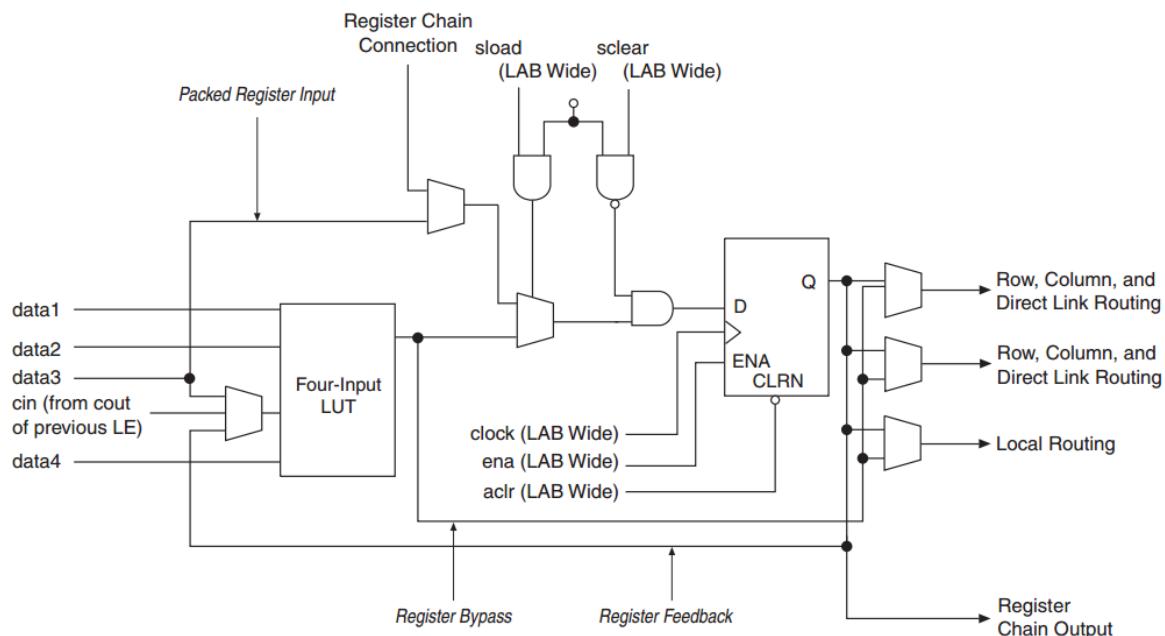
W układach rodziny *Cyclone III* bloki logiczne (*logic array blocks*) składają się z 16 elementów logicznych oraz bloku kontrolnego. Blok logiczny (LE) jest najmniejszą jednostką strukturalną w układzie FPGA rodziny *Cyclone III*.

Każdy element logiczny posiada 4 wejścia oraz czterowejściową tablicę LUT (*look-up table*), rejestr i logikę wyjściową. Jest to więc generator, na którym można zaimplementować każdą funkcję 4 zmiennych logicznych.

Rejestr, w każdym bloku logicznym, może zostać skonfigurowany do działania, jako przerzutnik następujących typów: D, T, JK lub SR. Każdy z rejestrów posiada sygnał danych, zegarowy, sygnał aktywacji wejścia zegarowego oraz wejście kasujące jego aktualny stan.

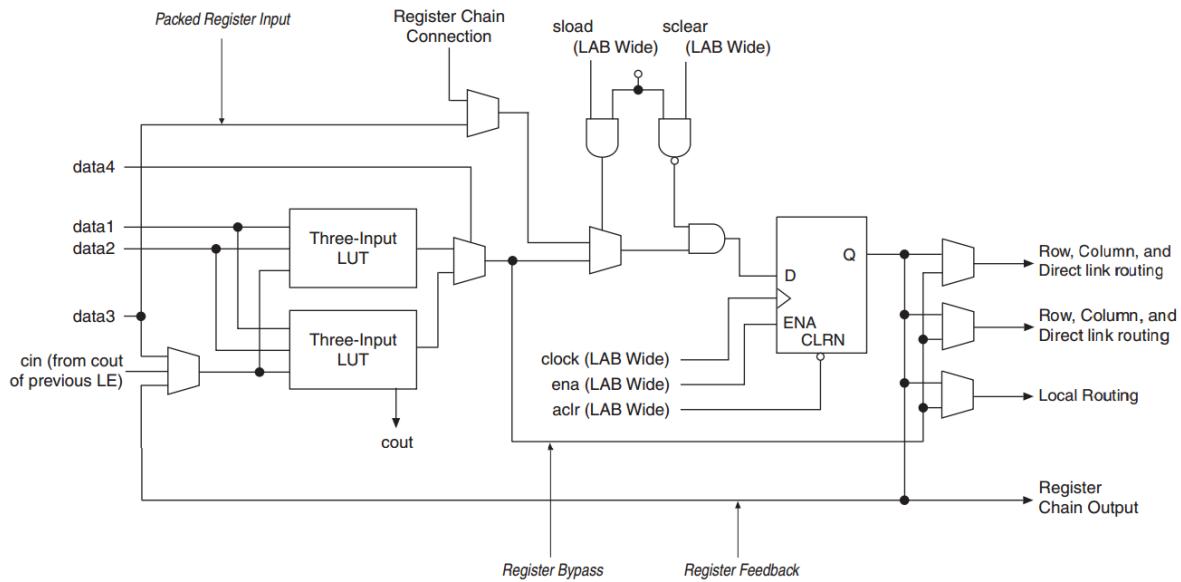
Elementy logiczne w układach *Altera Cyclone III* mogą pracować w jednym z dwóch dostępnych trybów: normalnym (*normal mode*) i arytmetycznym (*arithmetic mode*). Każdy z nich inaczej używa zasoby elementu logicznego. Oprogramowanie *Quartus II* automatycznie wybiera odpowiedni tryb dla popularnych funkcji, takich jak liczniki, sumatory, układy odejmujące.

Tryb normalny jest odpowiedni dla podstawowych aplikacji logicznych i funkcji kombinacyjnych. W tej konfiguracji, 4 wejścia z bloku logicznego kierowane są do wejść jednostki LUT.



Rys. 3. Element logiczny układu Cyclone II w trybie normalnym [1]

Tryb arytmetyczny jest wykorzystywany do tworzenia sumatorów, liczników, akumulatorów i komparatorów. Element logiczny w trybie arytmetycznym implementuje 2-bitowy pełny sumator i prosty łańcuch przeniesień i zapożyczeń. Kompilator programu *Quartus II* automatycznie tworzy logikę tego łańcucha podczas przetwarzania [1].



Rys. 4. Element logiczny układu Cyclone II w trybie arytmetycznym [1]

1. 2. 2. Bloki pamięci w rodzinie układów *Cyclone III*

Układy FPGA rodziny Cyclone III posiadają bloki pamięci *M9K*, które mogą spełniać różne funkcje, takie jak pamięć o swobodnym dostępie (RAM), rejesty przesuwne, pamięć tylko do odczytu (ROM) oraz kolejki FIFO. Bloki *M9K* mają pojemność 8192 bitów (9216 bitów wliczając bity parzystości) i mogą pracować w jednej z kilku konfiguracji, obejmującej następujące szerokości magistrali danych: 1, 2, 4, 8, 9, 16, 18, 32 i 36 bitów.

Bloki pamięci w układach *Cyclone III* posiadają sygnały kontrolne między innymi umożliwiające włączenie odczytu, zapisu oraz aktywację wejścia zegarowego. Do odczytu i zapisu mogą zostać użyte osobne sygnały taktujące, a więc zapis może przebiegać z inną częstotliwością niż odczyt. Możliwe jest także zastosowanie trybu *Dual-Port*, który pozwala na jednoczesny odczyt i zapis do pamięci dla różnych lokalizacji [1].

Dzięki tym funkcjom pamięci układu *Cyclone III*, wygodne jest przeprowadzenie wielu współbieżnych operacji, a także operacji wymagających różnych częstotliwości taktowania dla odczytu i zapisu (np. wyświetlanie grafiki przy użyciu wyjścia VGA).

1. 2. 3. 18-bitowe jednostki mnożące w układzie FPGA *Cyclone III*

Układ FPGA *Cyclone III* posiada dedykowane jednostki, które mogą wykonywać mnożenie liczb bez użycia dodatkowych elementów logicznych. Mogą one działać w jednym z dwóch trybów: mnożnika 18 x 18 lub dwóch mnożników 9 x 9 bitów. Dla liczb większych niż 18 bitów, oprogramowanie *Altera Quartus II* automatycznie łączy ze sobą bloki układów mnożących. Nie ma ograniczenia wielkości czynników, jednak proces mnożenia przebiega wolniej dla większych liczb.

Oprócz dedykowanych bloków mnożników, można wykorzystać programowe układy mnożące oparte o bloki pamięci M9K i tablice LUT elementów logicznych [1].

1. 2. 4. Pętle sprzężenia fazowego w układach rodziny *Cyclone III*

Pętle sprzężenia fazowego (Phase Locked Loop – PLL) są wykorzystywane do uzyskania wymaganej częstotliwości taktowania z podstawowego sygnału zegarowego. Układ FPGA Cyclone III posiada maksymalnie cztery jednostki PLL. Każda z nich ma 5 wyjść, które mogą generować różne, jednak zależne od siebie częstotliwości. Oprogramowanie *Quartus II* może automatycznie dobrać odpowiednie ustawienia układu PLL do generowania zadanej częstotliwości sygnału wyjściowego [1].

1. 3. 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.



Rys. 5. Układ FPGA Cyclone III na płytce modułu SNF-0

Wydany 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 [2].

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 [2].

2. Studium wykonalności

W rozdziale drugim przedstawiono wymagania postawione przed projektem i określono jego cele. W dalszej części przeprowadzono analizę ryzyka i propozycje rozwiązań potencjalnych problemów.

2. 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 projektu było stworzenie i przetestowanie przykładowej implementacji procesora, zapisanej w języku VHDL, który miał prezentować możliwości modułu.

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.

2. 2. Analiza ryzyka

Poniższe tabele przedstawiają analizę ryzyka. Pierwsza z nich opisuje nazwę ryzyka wraz z prawdopodobieństwem jego wystąpienia w skali od 1 do 10 oraz jego skutki i określenie poziomu szkód w dziesięciostopniowej skali.

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	Rzyko uszkodzenia lub niepoprawnego działania elementów modułu	8
7	Niewłaściwy dobór parametrów płytka PCB: grubości laminatu, wielkości otworów, odstępów między ścieżkami, wielkości przelotek, grubości ścieżek	4	Mała odporność mechaniczna płytka, zakłócenia, duże opóźnienia w transmisji sygnału, utrudniony montaż elementów	5
8	Zbyt niska jakość układu zasilania (wystąpienie zakłóceń, spadków napięcia pod obciążeniem)	4	Mozliwość 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

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.

3. Metodyka pracy

W poniższym rozdziale przedstawiono opis przyjętej metodyki oraz opisano narzędzia wykorzystywane w procesie tworzenia projektu inżynierskiego, a także przybliżono przebieg każdej z iteracji.

3. 1. Opis zastosowanej metodyki

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

3. 2. Narzędzia wykorzystywane w procesie tworzenia modułu i projektu procesora

- Altium Designer – program posłużył do zaprojektowania schematu modułu SNF-0 i wykonania obwodu drukowanego.
- Altera Quartus II – główne środowisko wykorzystane do projektowania przykładowego procesora wyświetlającego grafikę 3D zbudowaną z linii. Narzędzie to służyło zarówno, jako edytor kodu VHDL, jak również kompilator i programator.
- Sigasi; - edytor kodu VHDL i Verilog oparty o środowisko Eclipse. Wykorzystano go do pisania kodu VHDL modułów projektu i do refactoringu. Ścisła integracja z oprogramowaniem *Altera Modelsim* pozwoliła na zaawansowaną analizę napisanego kodu i jego symulację w czasie rzeczywistym.
- Autodesk 3DS Max – program został wykorzystany do eksportu modeli 3D do struktury danych przechowywanej na układzie FPGA. Napisano eksporter w języku skryptowym *MaxScript*, który zapisuje geometrię sceny do późniejszego wykorzystania w projekcie procesora graficznego.
- GitHub – został wykorzystany do przechowywania plików projektu i zarządzania jego rozwojem i wersjami
- Google Drive – do przechowywania kopii zapasowych, tworzenia dokumentacji i prezentacji
- draw.io – strona ułatwiająca tworzenie różnego rodzaju diagramów, użyto przy tworzeniu dokumentacji

3. 3. Przebieg prac

Podrozdział przedstawia opis poszczególnych iteracji projektu inżynierskiego. Opisano każdy z czterech przyrostów, podano datę jego rozpoczęcia i czas trwania. Przybliżono także cele iteracji, szczegółowo opisano jej realizację i przedstawiono wyniki.

3. 3. 1. Pierwszy przyrost

Data rozpoczęcia: 4.03.2016

Długość trwania: 2 miesiące

Cele

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

Realizacja

Po uzgodnieniu wymagań dotyczących projektu inżynierskiego, rozpoczęto tworzenie ogólnej koncepcji modułu uruchomieniowego.

Podstawowym problemem był wybór układu FPGA, stanowiącego główny element modułu. Zapoznano się z ofertą firmy Altera odnośnie układów rodziny Cyclone. Wybrano jej trzecią wersję, ze względu na dobry stosunek ceny, do jakości, zgodność z wymaganiami stawianymi przed projektem oraz dostępność w hurtowniach elektronicznych.

Kolejną fazą tej iteracji było dobranie elementów peryferyjnych układu FPGA. Po przeanalizowaniu instrukcji rodziny układów Cyclone III i zapoznaniu się z podobnymi projektami, przystąpiono do tworzenia wstępnej wersji schematu i wyboru układów obsługi wejścia/wyjścia. Zakupiono programator zgodny z *Altera USB-Blaster* [7].

Tworzenie schematu rozpoczęto od umiejscowienia układu FPGA i podłączenia go do pamięci SSRAM, wykorzystując specjalne porty pamięci zewnętrznej dostępnych w układach rodziny *Cyclone III*. Wykorzystano wszystkie wyprowadzenia posiadające taką funkcję.

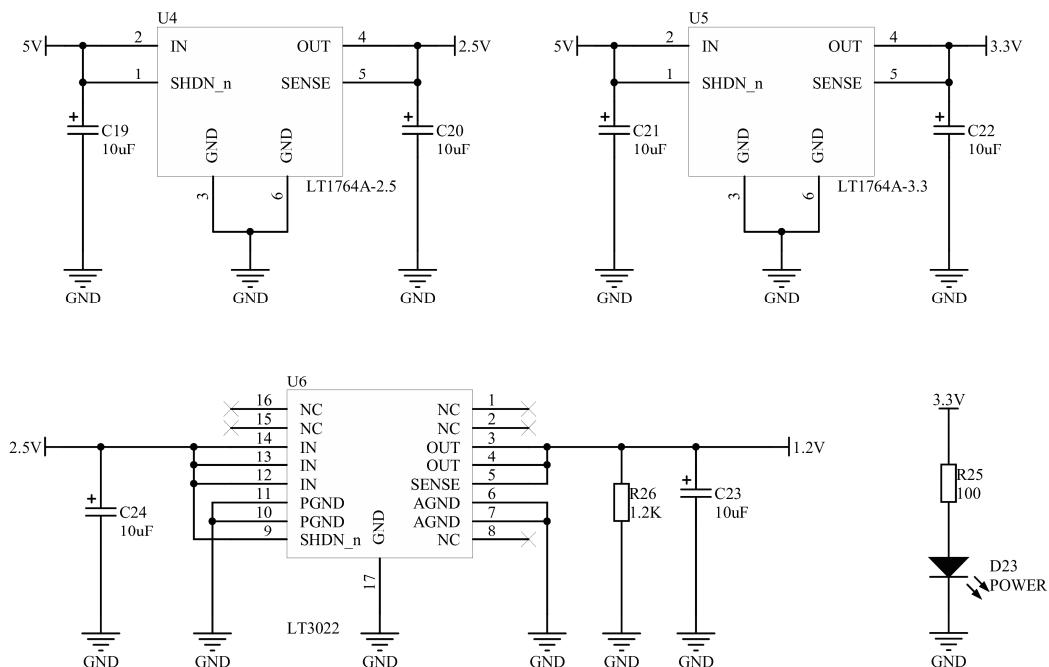
Kolejnym ważnym krokiem rysowania schematu było przygotowanie modułu VGA1. Wybór układu bufora obrazu *SSD1963* poprzedzony był dłuższym poszukiwaniem. Zintegrowany układ firmy *Solomon Systech* okazał się spełniać większość oczekiwanych [4].

Do zamiany 24-bitowej wartości koloru na sygnał analogowy zgodny ze standardem VGA, wykorzystano specjalizowany układ trójkanałowego przetwornika cyfrowo-analogowego firmy *Analog Devices* [6].

Ważnym aspektem projektu połączeń wyżej wymienionych układów z FPGA było takie wybranie wyprowadzeń, posiadających wymagane funkcje, aby podczas tworzenia obwodu drukowanego występuała jak najmniejsza liczba skrzyżowań.

Po raz pierwszej rysowano także schemat połączeń układu FPGA z portami wejścia/wyjścia (VGA2, GPIO, PS2, UART) oraz periferii służących do programowania układu *Cyclone III*.

Ostatnim ważnym elementem pracy nad schematem było przygotowanie układu zasilania. Wybrano stabilizatory liniowe o niskim spadku napięcia (*LDO*). W późniejszych iteracjach zamieniono je jednak na stabilizatory impulsowe. Rysunek 6 przedstawia początkowy schemat układu zasilania.



Rys. 6. Początkowa wersja układu zasilania

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.

3. 3. 2. Drugi przyrost

Data rozpoczęcia: 4.05.2016

Długość trwania: 2 miesiące

Cele

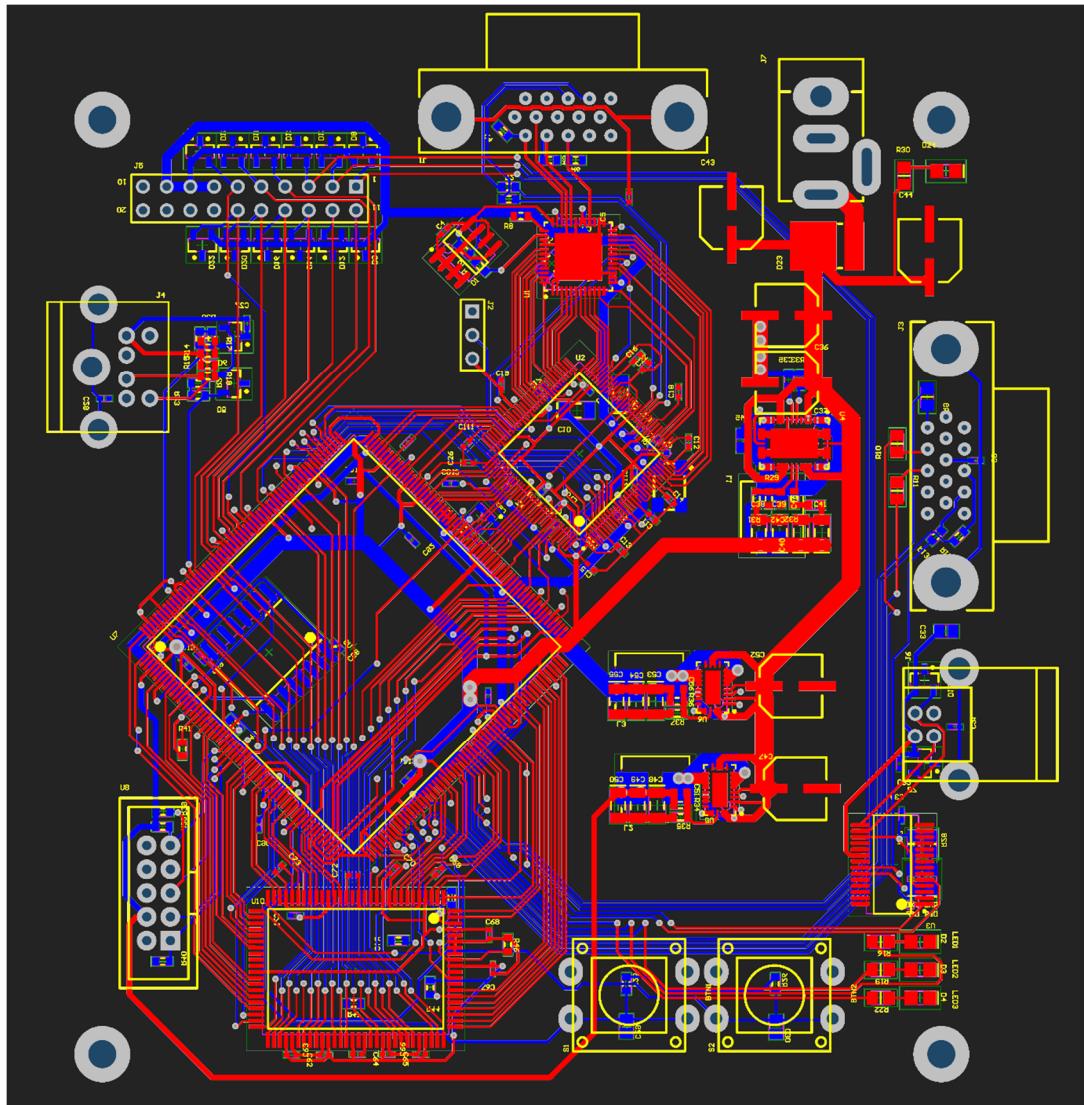
- Ostateczna wersja schematu modułu SNF-0
- Prototyp obwodu drukowanego
- Ostateczna lista elementów
- Zakup potrzebnych elementów elektronicznych

Realizacja

Kolejnym ważnym etapem realizacji projektu inżynierskiego było narysowanie obwodu drukowanego modułu SNF-0. Prace rozpoczęto zaraz po przygotowaniu wstępnego schematu. Jednak po dokładniejszej analizie wymagań napięciowych i prądowych modułu i konsultacji z Promotorem, postanowiono przebudować układ zasilania. Od sprawności podsystemu dostarczającego napięcia do układów modułu zależy stabilność ich pracy.

W drugim etapie projektu zostały zamówione niektóre elementy, takie jak: przetwornik cyfrowo-analogowy, stabilizatory (nieużyte w końcowej wersji modułu) i inne mniej istotne podzespoły.

Po przeprojektowaniu schematu układu zasilania i wprowadzeniu drobnych poprawek, przystąpiono do tworzenia obwodu drukowanego. Początkowym etapem tego procesu było zapoznanie się ze oprogramowaniem *Altium Designer* i zaczęto projektować prototyp PCB. Pozwolił on lepiej zidentyfikować wymagania i określić możliwości [10]. Rysunek 7 przedstawia prototyp obwodu drukowanego. W tym czasie przygotowano także bibliotekę, zawierającą większość projektów niestandardowych elementów.



Rys. 7. Prototyp obwodu drukowanego modułu SNF-0

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.

3. 3. 3. Trzeci przyrost

Data rozpoczęcia: 4.07.2016

Długość trwania: 3 miesiące

Cele

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

Realizacja

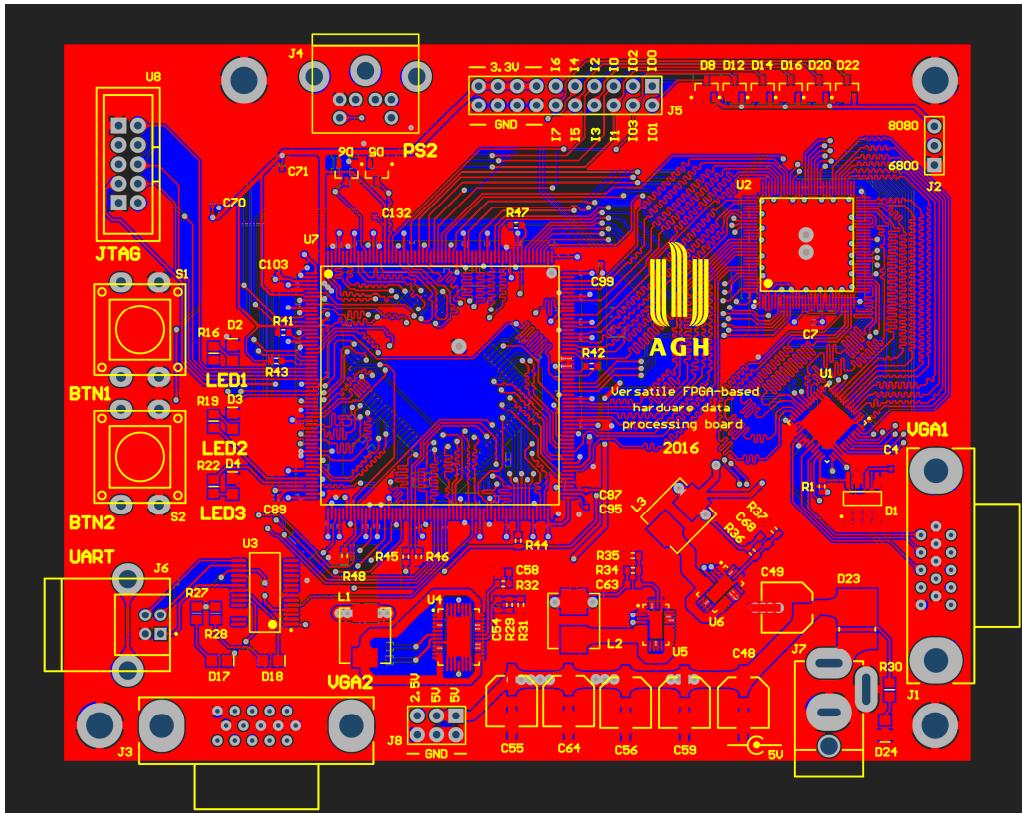
W trzecim przyroście, na podstawie wcześniej wykonanego prototypu, zaprojektowano ostateczną wersję obwodu drukowanego. Dzięki lepszemu poznaniu oprogramowania, możliwe było szybkie i bardziej adekwatne rozłożenie elementów.

Postanowiono zastosować technikę wyrównywania długości ścieżek (*length tuning*), która pozwoli na zmniejszenie błędów przesyłania danych przez ważne magistrale (między innymi połączenie FPGA i pamięci SSRAM oraz FPGA i bufora obrazu). Starano się także zredukować ogólną długość połączeń, zwłaszcza między istotnymi komponentami [10].

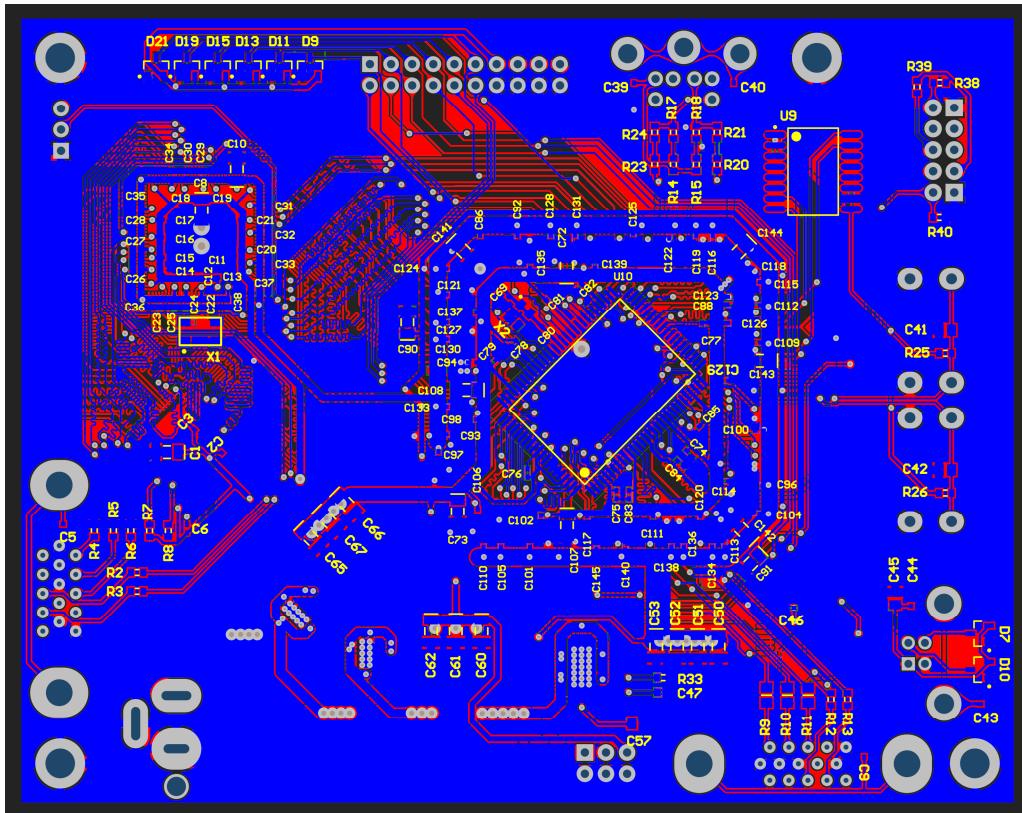
Rozłożenie elementów zostało poprawione. Dostęp do złącz został zoptymalizowany. Rozmiary płytki uległy nieznaczнемu zwiększeniu w stosunku do prototypu między innymi ze względu na rozrost obszaru zajmowanego przez połączenia z zastosowaną techniką *length tuning*.

Złącza i kontrolki zostały ułożone w sposób ułatwiający późniejsze używanie modułu SNF-0. Wykonano otwory montażowe, w których miały być umieszczone podpory mocujące obudowę. Poprawiono także układ i czytelność opisów, ułatwiających montaż i użytkowanie płytki.

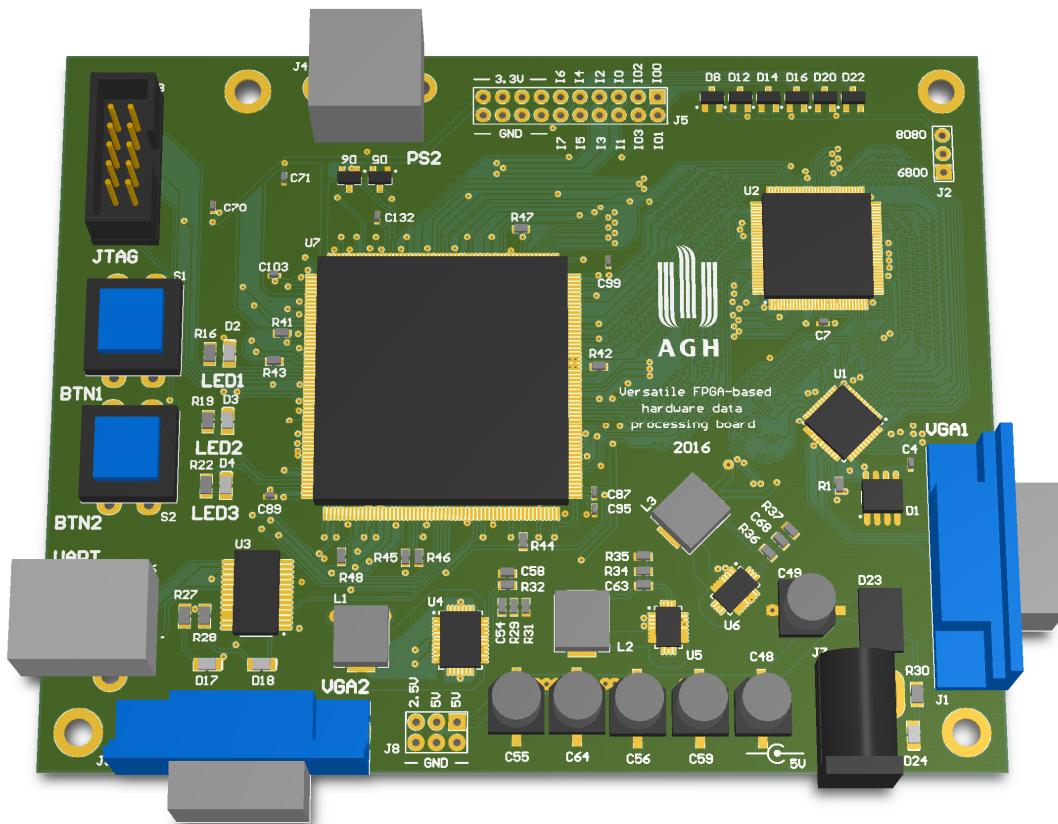
Projekt końcowej wersji obwodu drukowanego modułu SNF-0 przedstawiony został na rysunkach 8 i 9. Wizualizacja 3D płytki PCB zaprezentowana została na rysunku 10.



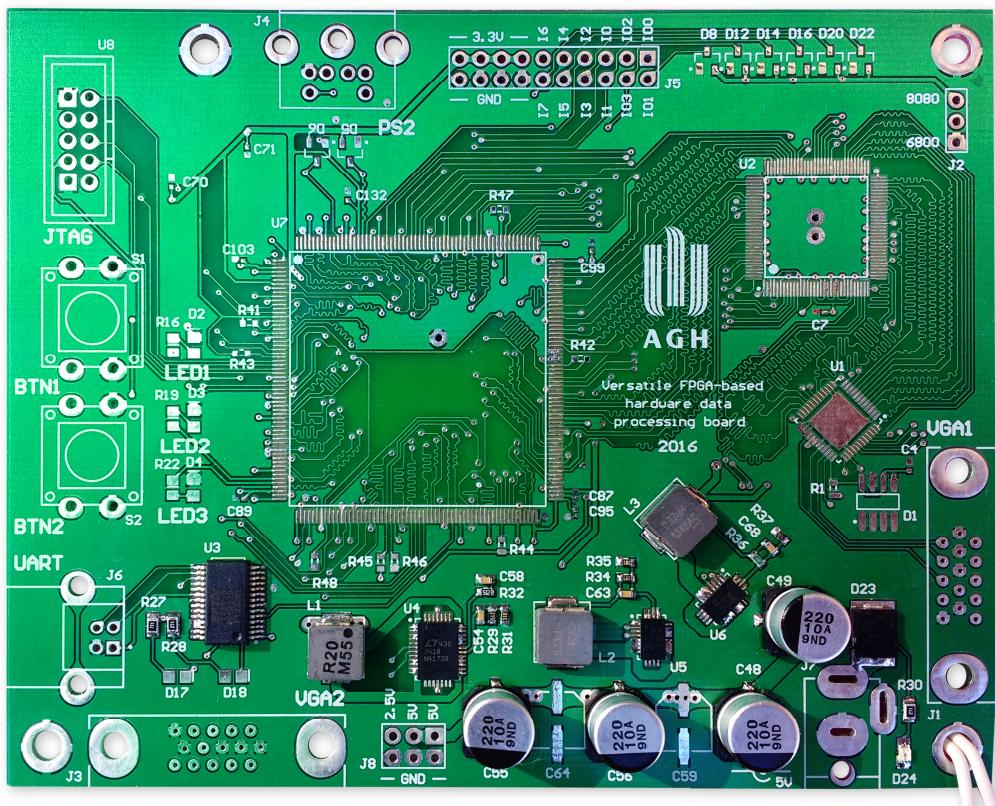
Rys. 8. Projekt obwodu drukowanego widziany z góry



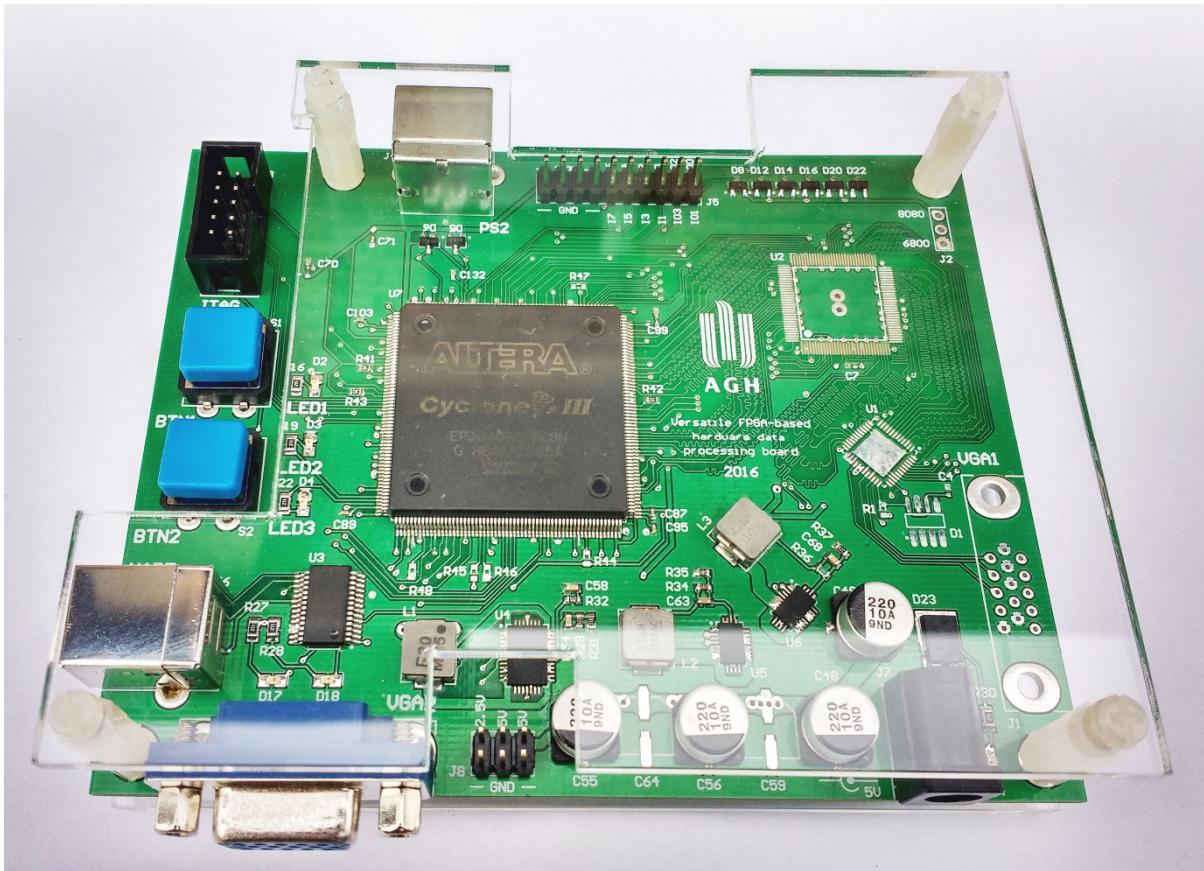
Rys. 9. Projekt obwodu drukowanego widziany z dołu



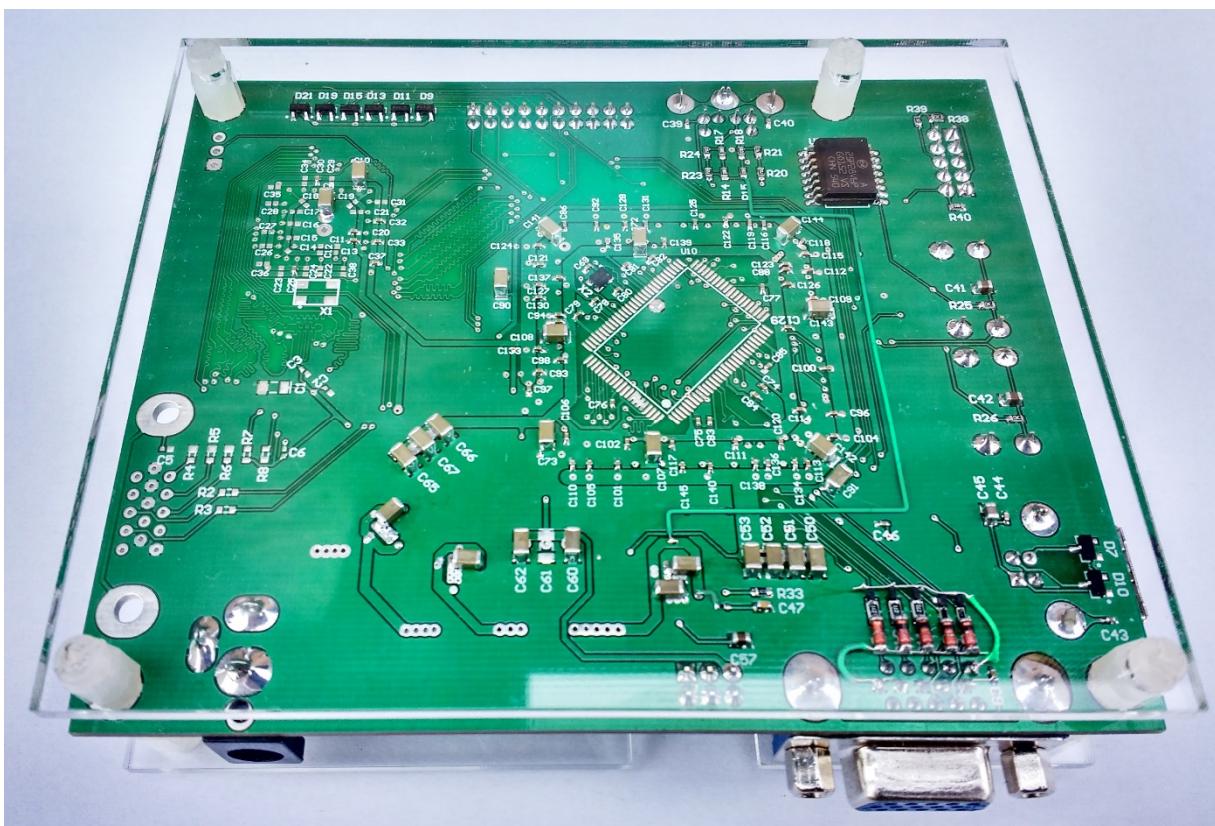
Rys. 10. Wizualizacja modułu SNF-0



Rys. 11. Moduł SNF-0 podczas montażu



Rys. 12. Gotowy moduł SNF-0 (widok od góry)



Rys. 13. Gotowy moduł SNF-0 (widok od dołu)

Kolejnym etapem trzeciej iteracji było wykonanie i uruchomienie modułu SNF-0. Wykonanie samej płytki PCB powierzone zostało firmie oferującej takie usługi. Po skompletowaniu wszystkich podzespołów, przystąpiono do montażu modułu. Zdjęcie 11 przedstawia proces montażu płytka PCB.

Na koniec etapu zaprojektowano i zlecono wycięcie prostej, dwuczęściowej obudowy ze szkła akrylowego, która została zamocowana przy użyciu plastikowych elementów montażowych.

Gotowy projekt pokazany został na zdjęciach 12 i 13.

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).

3. 3. 4. Czwarty przyrost

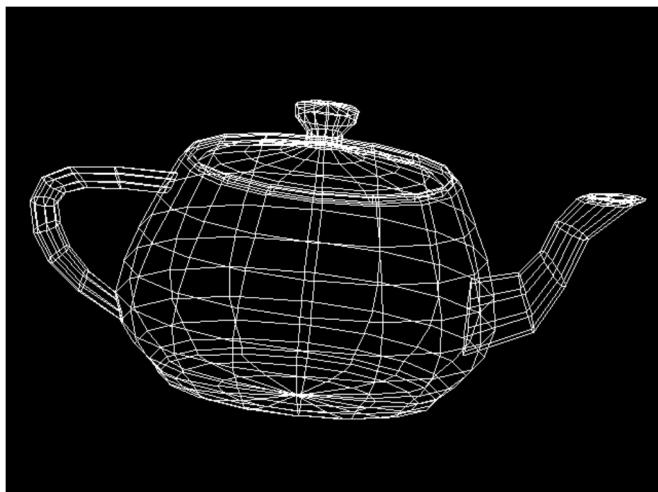
Data rozpoczęcia: 4.09.2016

Długość trwania: 3 miesiące

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

Realizacja



Rys. 14. Prototypowy program o funkcjonalności projektowanego procesora graficznego

można znaleźć w repozytorium projektu inżynierskiego (https://github.com/papciakk/FPGA_Board) w folderze *software/3D_processor_prototype*. Do jego komplikacji potrzebna jest biblioteka *SDL (Simple DirectMedia Layer)* oraz jej rozszerzenie *SDL_gfx*. Na rysunku 14 przedstawiono zrzut ekranu z omawianego programu.

Po uruchomieniu podstawowej wersji projektu specjalizowanego procesora graficznego na zestawie SNF-0, przystąpiono do rozszerzania jego funkcjonalności. Między innymi poprawiono sposób wgrywania modeli 3D do pamięci układu FPGA. Przygotowano także eksporter z programu *Autodesk 3DS Max*, który pozwala wyeksportować wczytaną scenę do pliku *.vhd zawierającego strukturę danych wykorzystywaną przez projekt.

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.

Pierwszym etapem czwartej iteracji było przygotowanie zestawu jednostek projektowych języka VHDL odpowiedzialnych za obsługę dostępnych peryferiów. Podstawowe wersje sterowniki poszczególnych portów były implementowane i testowane zaraz po zmontowaniu elementów modułu.

Jako podstawę przykładowego procesora prezentującego działanie modułu SNF-0, wykorzystano przebudowany projekt z przedmiotu *Technika Mikroprocesorowa 2*.

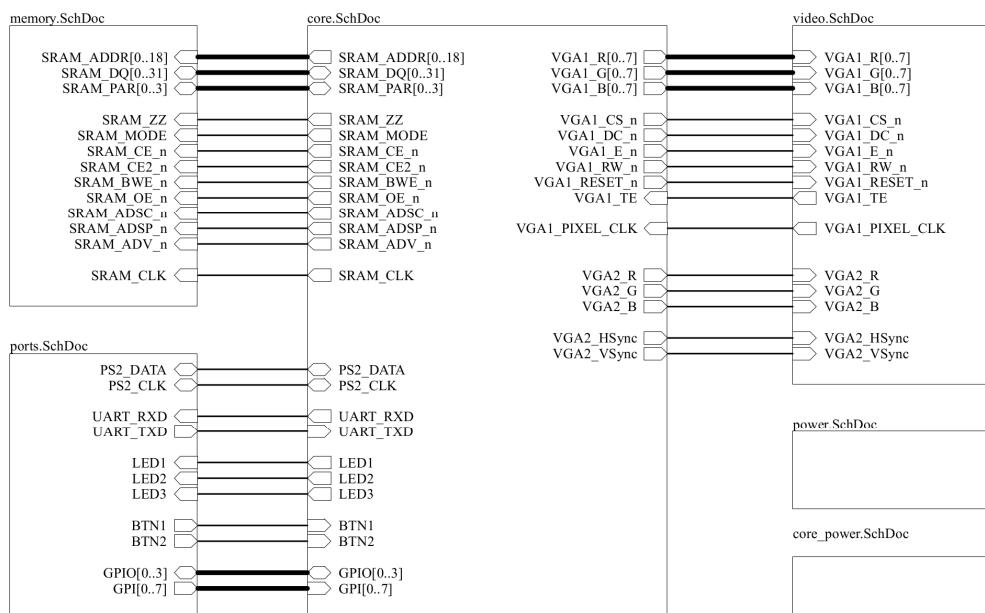
Przed przystąpieniem do tworzenia specjalizowanego procesora graficznego, przygotowano prototyp w formie programu komputerowego, który posiadał taką samą funkcjonalność. Kod programu napisany w języku C++

4. Opis szczegółów implementacyjnych

W rozdziale czwartym omówiono poszczególne elementy schematu modułu SNF-0. Przedstawiono funkcje pełnione przez podzespoły wykorzystane do jego budowy i opisano możliwości, które ze sobą niosą. W dalszej części rozdziału opisano wykonanie obwodu drukowanego modułu i przygotowanie procesora graficznego.

4. 1. Schemat modułu SNF-0

Głównym elementem modułu SNF-0 jest układ FPGA *Altera Cyclone III*. Do niego podłączone są wyrowadzenia 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.



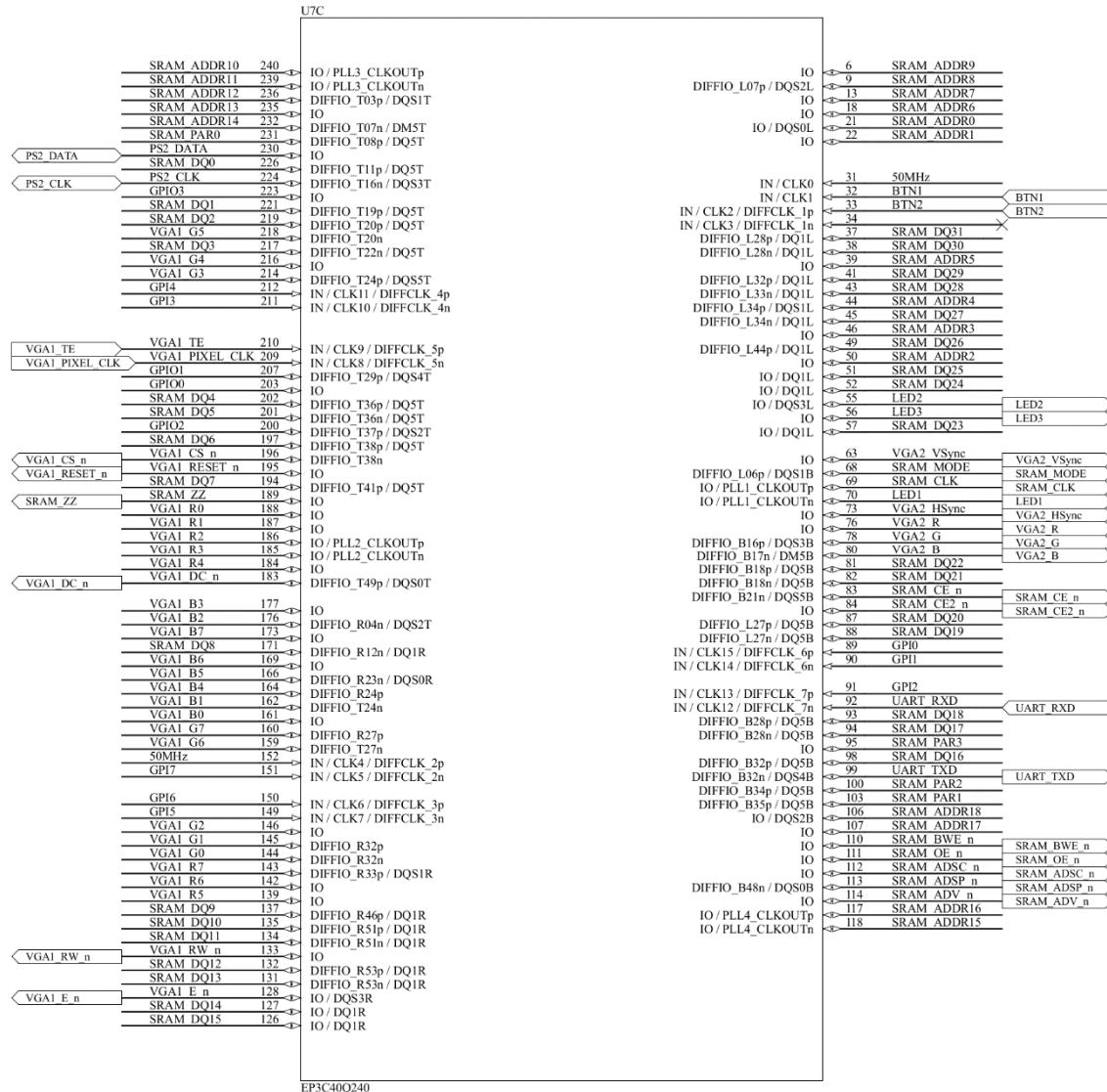
Rys. 15. Podział i połączenie dokumentów schematu modułu SNF-0

Schemat obwodu drukowanego zaprojektowano w programie *Altium Designer*. Jest on podzielony na 6 bloków (rys. 6). Plik *memory.SchDoc* przedstawia połączenia pamięci SRAM z układem FPGA oraz jej otoczenie. *core.SchDoc* zawiera schemat połączeń układu FPGA z innymi elementami modułu SNF-0 oraz układ programowania, natomiast *core_power.SchDoc* jego połączenia z napięciami zasilającymi. W pliku *ports.SchDoc* przedstawiono połączenie portów wejścia/wyjścia, PS2, portu szeregowego i przycisków oraz diod LED. Plik *video.SchDoc* prezentuje sposób połączenia portów VGA oraz układ bufora obrazu i przetwornika cyfrowo-analogowego. W dokumencie *power.SchDoc* znajduje się schemat układu zasilania.

W poniższych podrozdziałach przedstawiono opis poszczególnych obwodów modułu SNF-0. Pełny zestaw schematów znajduje się na nośniku CD dołączonym do dokumentacji oraz w repozytorium projektu pod adresem: https://github.com/papciakk/FPGA_Board w folderze *doc*, w pliku *schematic.pdf*.

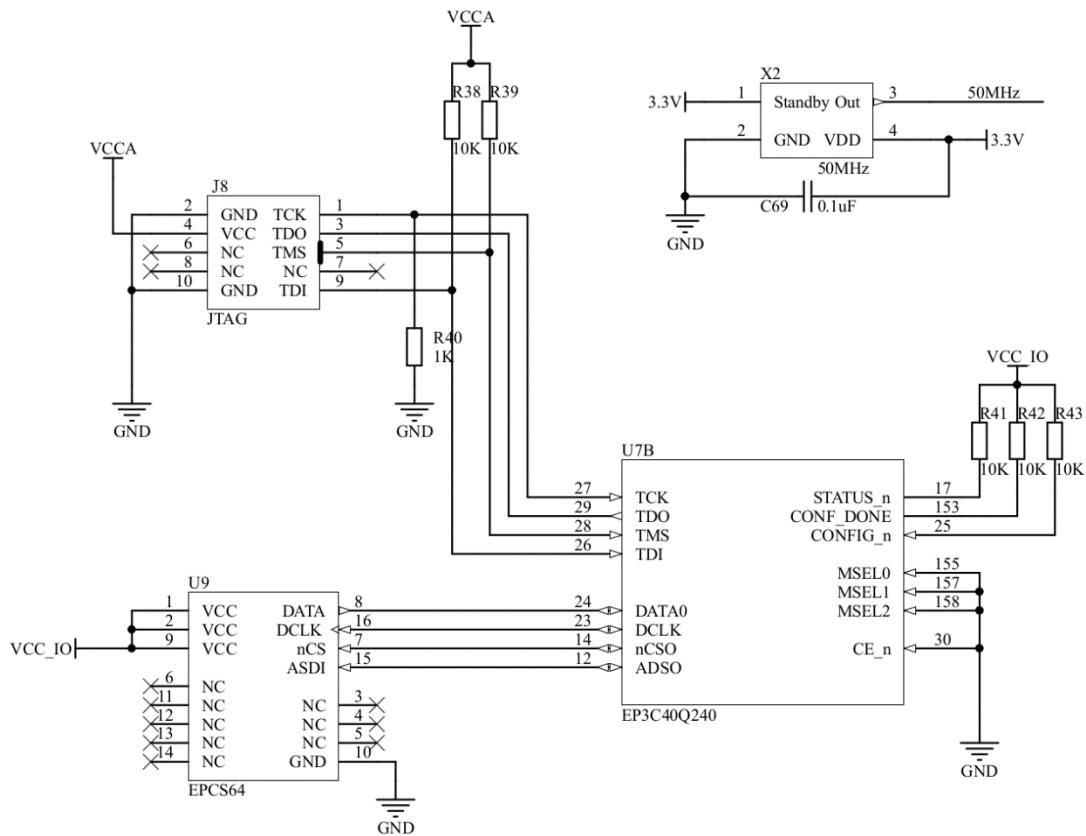
4. 1. 1. Układ FPGA i jego otoczenie

Rysunek 16 prezentuje podłączenie portów wejścia/wyjścia układu FPGA *Cyclone III* do poszczególnych podzespołów modułu.



Rys. 16. Schemat podłączenia portów wejścia/wyjścia układu FPGA

Układ FPGA modułu SNF-0 programowany jest za pomocą programatora zgodnego z Altera USB-Blaster [7]. Konfiguracja wykonywana jest w trybie JTAG. Domyślnie konfiguracja układu FPGA zanika po wyłączeniu zasilania. Moduł SNF-0 daje możliwość wykorzystania układ szeregowej pamięci EEPROM, który może ją przechowywać (w obecnym stanie nie jest wykorzystywana).

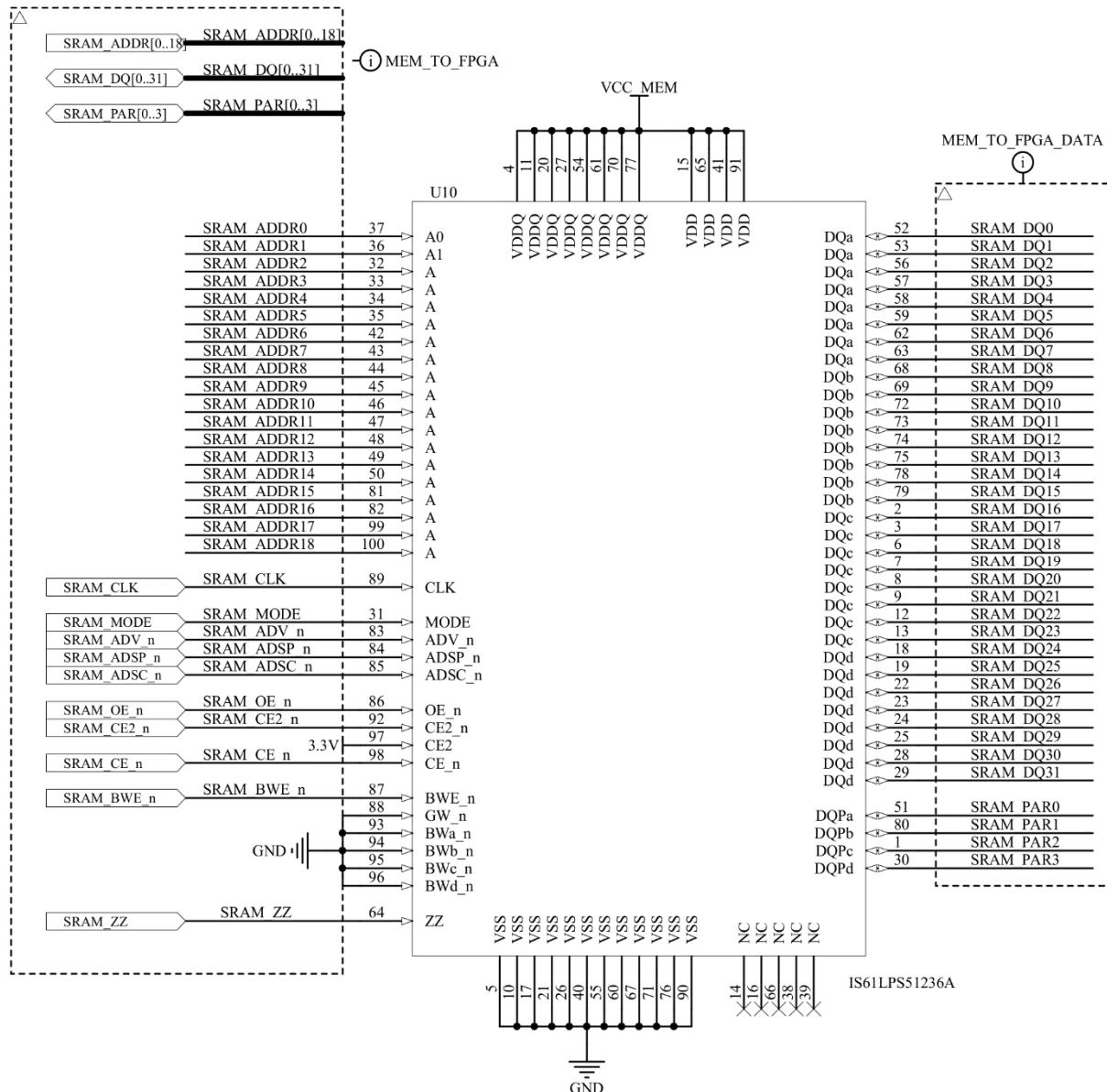


Rys. 17. Schemat układu programowania i taktowania modułu SNF-0

Na rysunku 17 pokazano schemat układu programowania i oscylator generujący główny sygnał zegarowy modułu SNF-0. Złącze programatora zgodnego z USB-Blaster zaznaczono symbolem J8. Układ U9 to szeregowa pamięć EEPROM mogąca przechowywać konfiguracje układu FPGA po odłączeniu zasilania [11]. U7B to wyprowadzenia układu FPGA odpowiedzialne za programowanie. Piny *MSEL[0..2]* konfigurują tryb programowania układu *Cyclone III*. Na schemacie symbolem X2 oznaczony został oscylator *MEMS* generujący sinusoidalny sygnał taktujący o częstotliwości 50MHz, który podłączony jest do wejść zegarowych układu FPGA.

4. 1. 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.

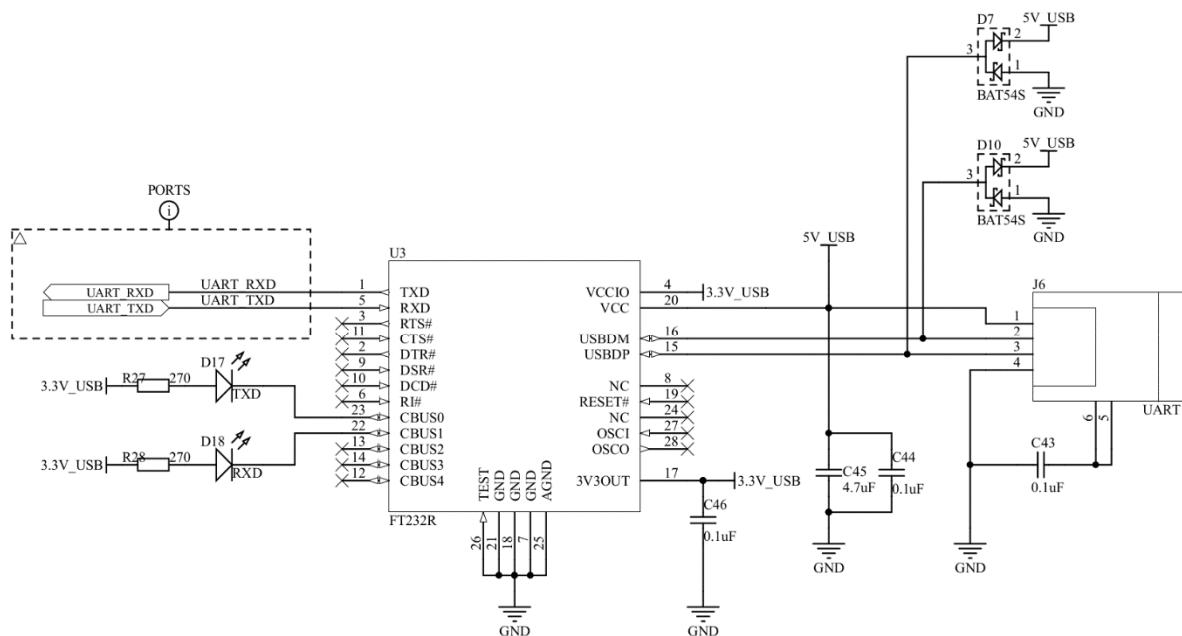


Rys. 18. Schemat podłączenia pamięci SSRAM do układu FPGA

Układ ten oznaczony jest na schemacie (rysunek 18) symbolem U10. Pamięć SRAM 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 U10 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 [5].

4. 1. 3. Port szeregowy

Podstawowym portem komunikacyjnym wykorzystywany 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.



Rys. 19. Schemat interfejsu portu szeregowego

Układ U3 (*FT232R* firmy *FTDI Chips*), przedstawiony na rysunku 19 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.

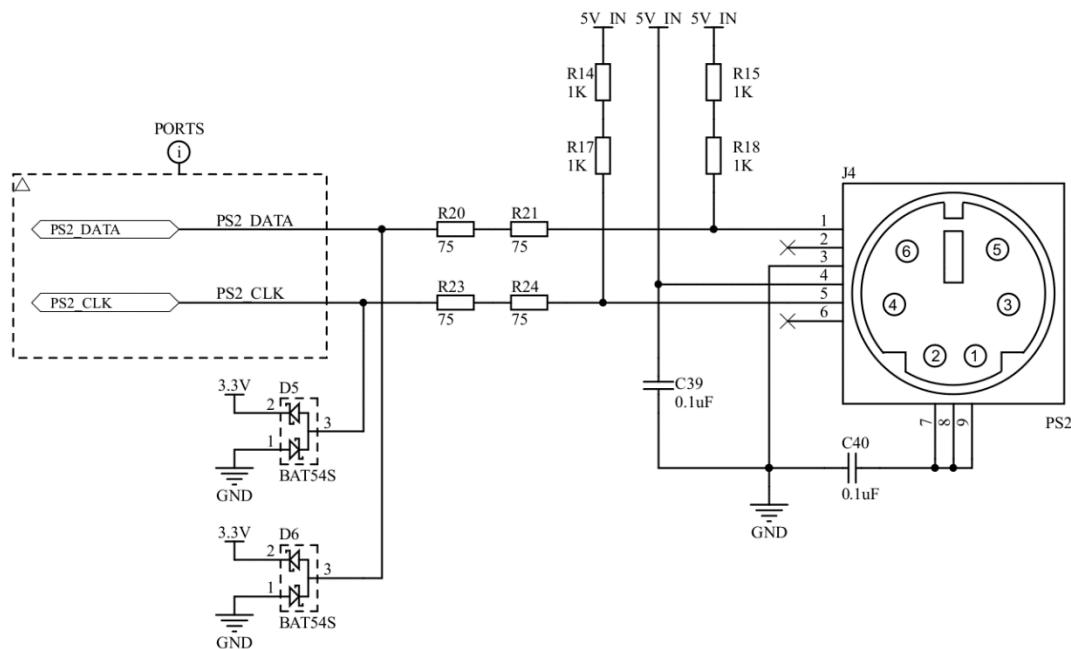
Układ firmy *FTDI Chips* oferuje także 5 konfigurowalnych portów ogólnego przeznaczenia. W opisywanej aplikacji podłączone są do diod LED, które wskazują trwającą transmisję (D17) i odbiór (D18). Interfejs USB do portu szeregowego oraz diody wskazujące transmisję danych zasilane są bezpośrednio z portu USB przy użyciu wewnętrznego stabilizatora, który obniża napięcie z 5V do 3.3V. Wykorzystano proste zabezpieczenie przy użyciu diod D7 i D10, które redukują skoki napięcia w razie przepięć elektrostatycznych lub pojawiennia 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 ustalenie prezentowanej nazwy interfejsu, czy też konfiguracja portów ogólnego przeznaczenia [3].

4. 1. 4. Port PS2

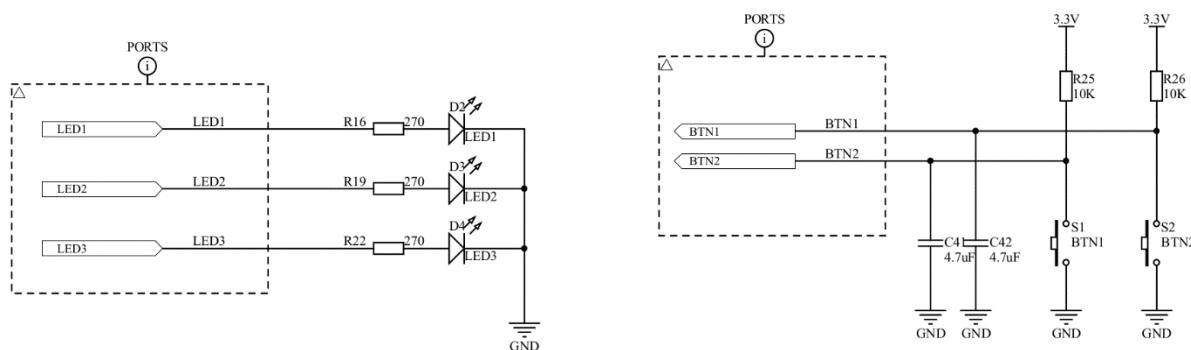
Rysunek 20 przedstawia schemat podłączenia portu PS2 do układu FPGA, które zostały zabezpieczone przed skokami napięcia za pomocą diod D5 i D6. Zastosowanie tego portu umożliwia łatwe podłączenie do modułu SNF-0 klawiatury lub myszki zgodnej ze standardem PS2.



Rys. 20. Schemat podłączenia portu PS2

4. 1. 5. Przyciski i diody LED

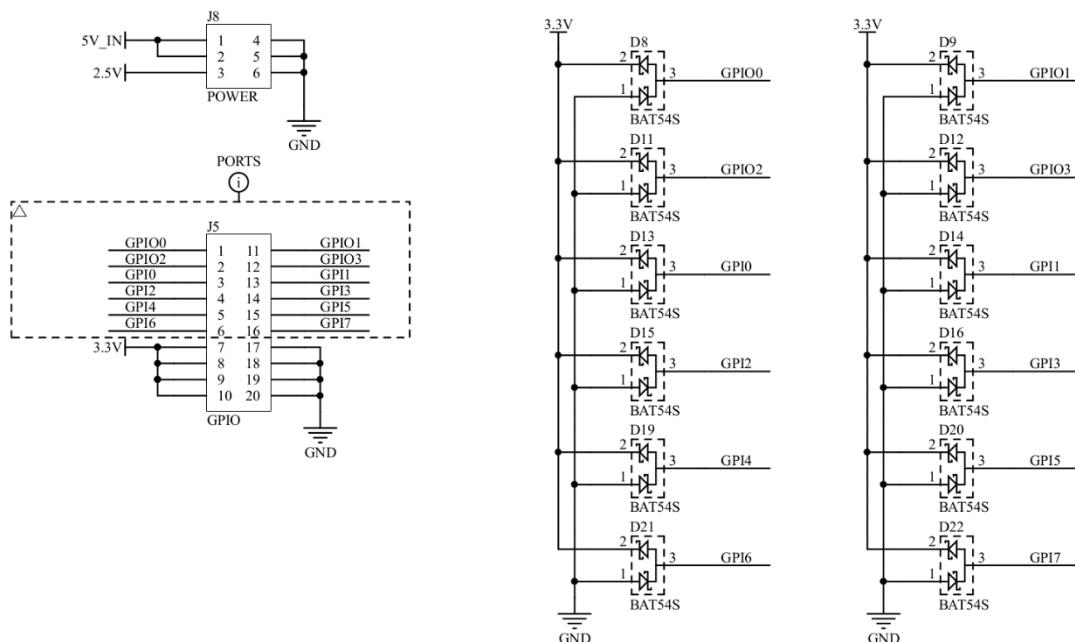
Elementami zestawu SNF-0 są 3 diody LED oraz 2 przyciski, posiadające prosty filtr redukujący niepożądane efekty związane z drgiem styków. Zarówno diody, jak i przyciski mogą zostać wykorzystane w projektach w dowolny sposób, pozwalając na interakcję z użytkownikiem. Schemat podłączenia omawianych elementów pokazano na rysunku 21.



Rys. 21. Schemat podłączenia diod LED i przycisków do układu FPGA

4. 1. 6. Porty wejścia/wyjścia ogólnego przeznaczenia i wyprowadzenia napięć zasilających

Dostępne są 4 porty wejścia/wyjścia i 8 portów wejściowych ogólnego przeznaczenia (przedstawione na rysunku 22). 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.



Rys. 22. Schemat portów wejścia/wyjścia ogólnego przeznaczenia

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.

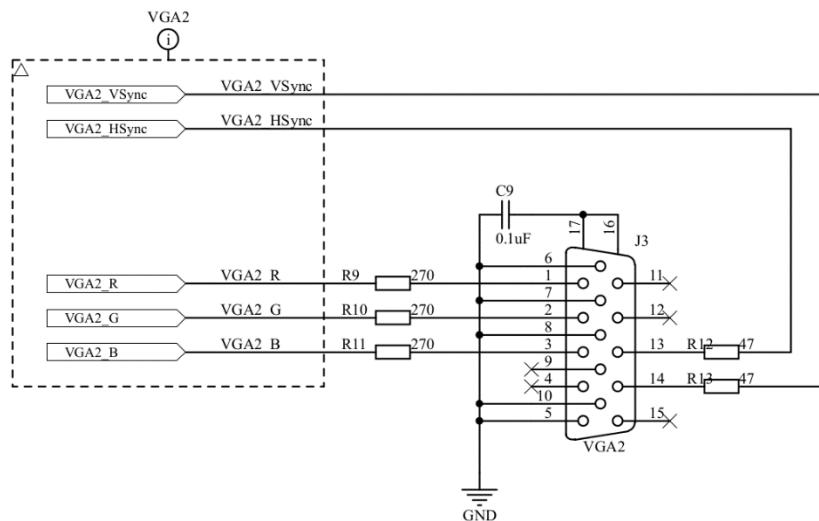
Dzięki stosunkowo dużej szybkości kontrolera wejścia/wyjścia w układzie *Altera Cyclone III*, możliwe jest używanie dostępnych wyprowadzeń, jako portów sygnałowych standardu SPI lub podłączenie do nich zewnętrznego modułu Ethernet.

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.

4. 1. 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 (przykładowe rozdzielczości możliwe do uzyskania przedstawiono w tabeli 1). Wymagane jest, aby częstotliwość generowania pikseli była możliwa do uzyskania za pomocą układu FPGA modułu SNF-0. Port VGA2 posiada 3 jednabitowe sygnały kolorów oraz sygnały synchronizacji poziomej i pionowej.

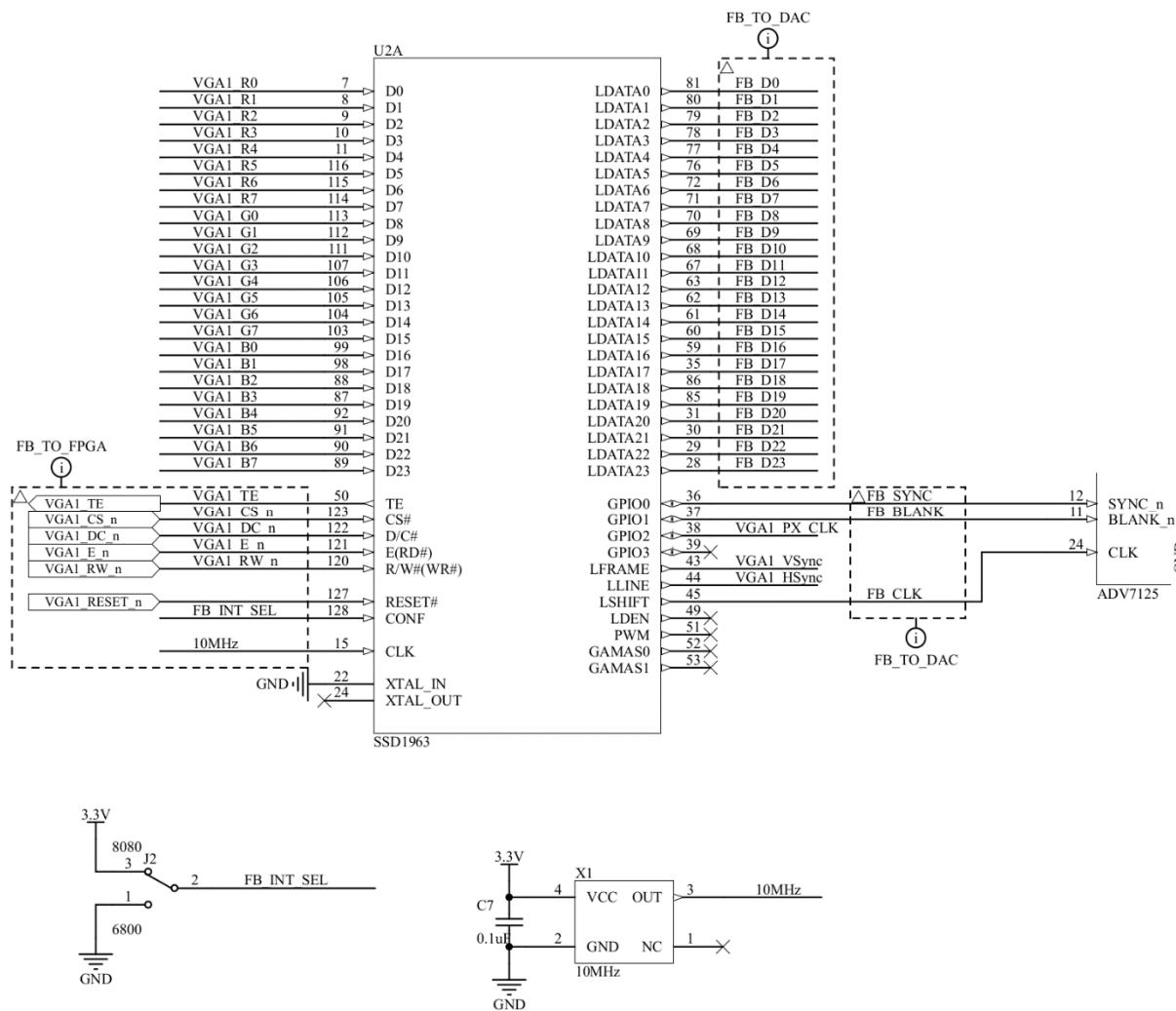


Rys. 23. Schemat wyjścia VGA2

Na rysunku 23 zaprezentowano połączenie wyjścia VGA2 do układu FPGA.

4. 1. 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.



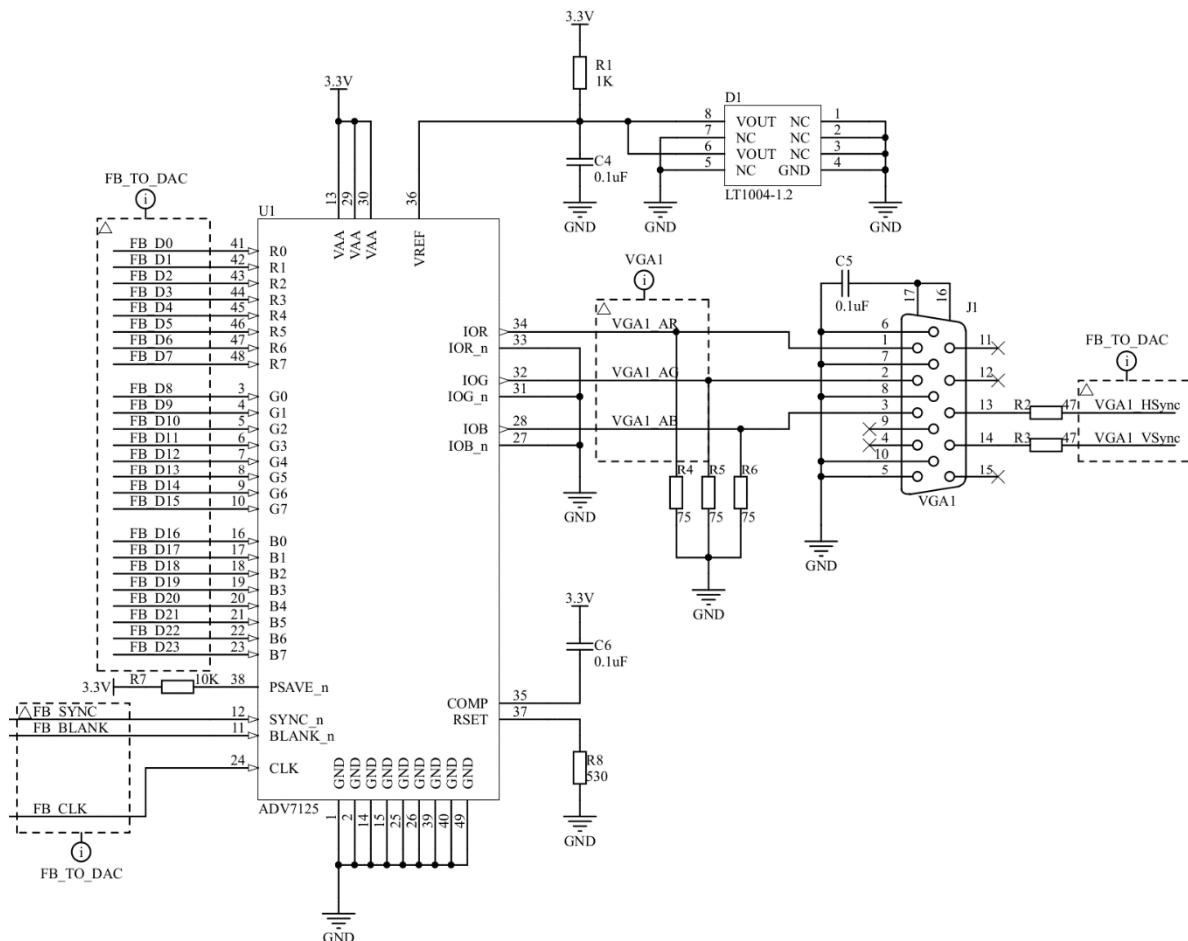
Rys. 24. Schemat połączeń układu bufora obrazu i jego otoczenia

Układ U2A (SSD1963 firmy Solomon Systech), przedstawiony na rysunku 24, jest buforem obrazu o pojemności 1215KB ze zintegrowanym kontrolerem wyświetlania. Dostępna pamięć pozwala na przechowanie obrazu o rozdzielcości 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). Tryb interfejsu ustawiany jest za pomocą zworki J2. W otoczeniu układu kontrolera obrazu znajduje się oscylator kwarcowy (X1) o częstotliwości 10MHz, który stanowi dokładniejszą alternatywę do wbudowanego generatora sygnału zegarowego. SSD1963 wymaga dwóch różnych napięć: 1.2V -do zasilania logiki układu i 3.3V dla sygnałów wejścia/wyjścia.

Oprócz podstawowej funkcjonalności, omawiany element posiada kilka przydatnych funkcji pozwalających na manipulację wyświetlonym 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 [4].



Rys. 25. Schemat wyjścia VGA1 i przetwornika cyfrowo-analogowego

Kolejnym ważnym elementem podsystemu wyświetlania VGA1 jest układ 3-kanałowego przetwornika cyfrowo-analogowego U1 (ADV7125 firmy *Analog Devices*). Jest to specjalizowany konwerter przeznaczony do zastosowań związanych z wyświetlaniem obrazu o trzech składowych koloru [6].

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 poziomie napięcia, który steruje jasnością aktualnie wyświetlonego subpiksela.

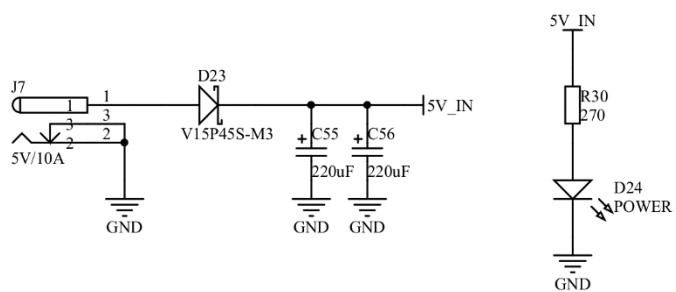
Wykorzystano zewnętrzny układ napięcia referencyjnego 1.235V, którego głównym elementem jest dioda Zenera D1 o dużej dokładności.

Podsystem wyświetlania obrazu ze złączem VGA1 pozwala na tworzenie projektów wyświetlających zaawansowaną grafikę w 24-bitowej głębi kolorów. Wadą tego rozwiązania jest jednak maksymalna rozdzielcość możliwa do uzyskania, wynosząca 640 x 480 pikseli. Schemat omawianego obwodu pokazano na rysunku 25.

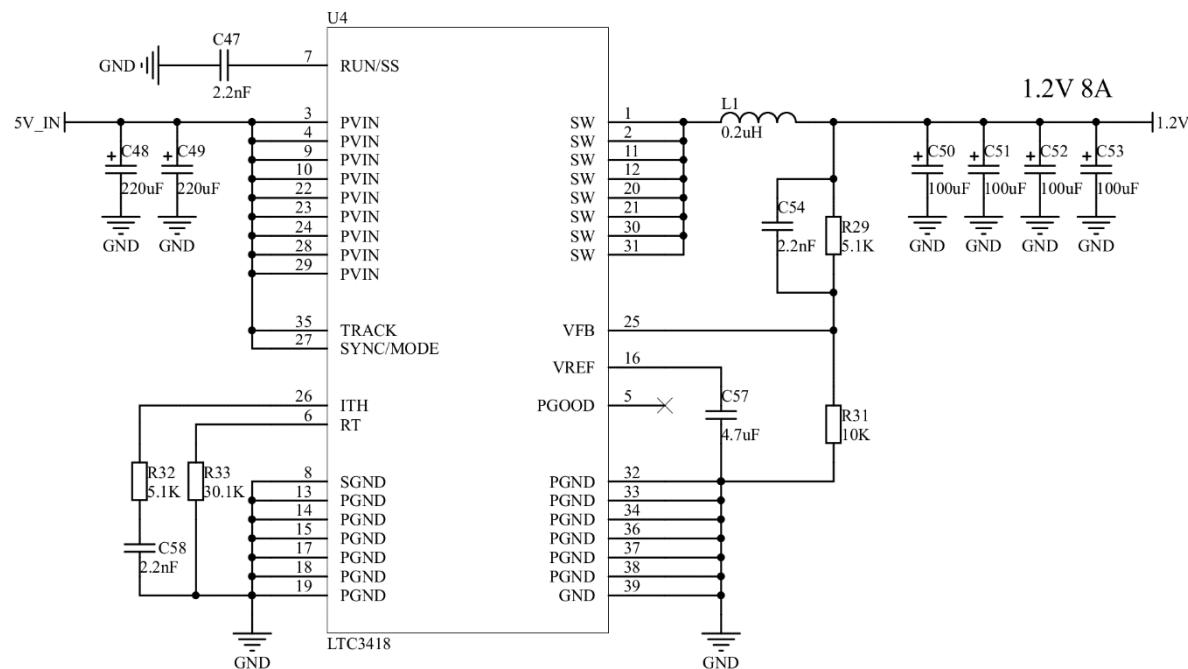
4. 1. 9. Moduł zasilający

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 bieguności. Dioda LED D24 wskazuje obecność napięcia zasilania. Rysunek 26 pokazuje schemat podłączenia diody Zenera i kontrolki zasilania.



Rys. 26. Schemat złącza zasilania i elementów peryferyjnych

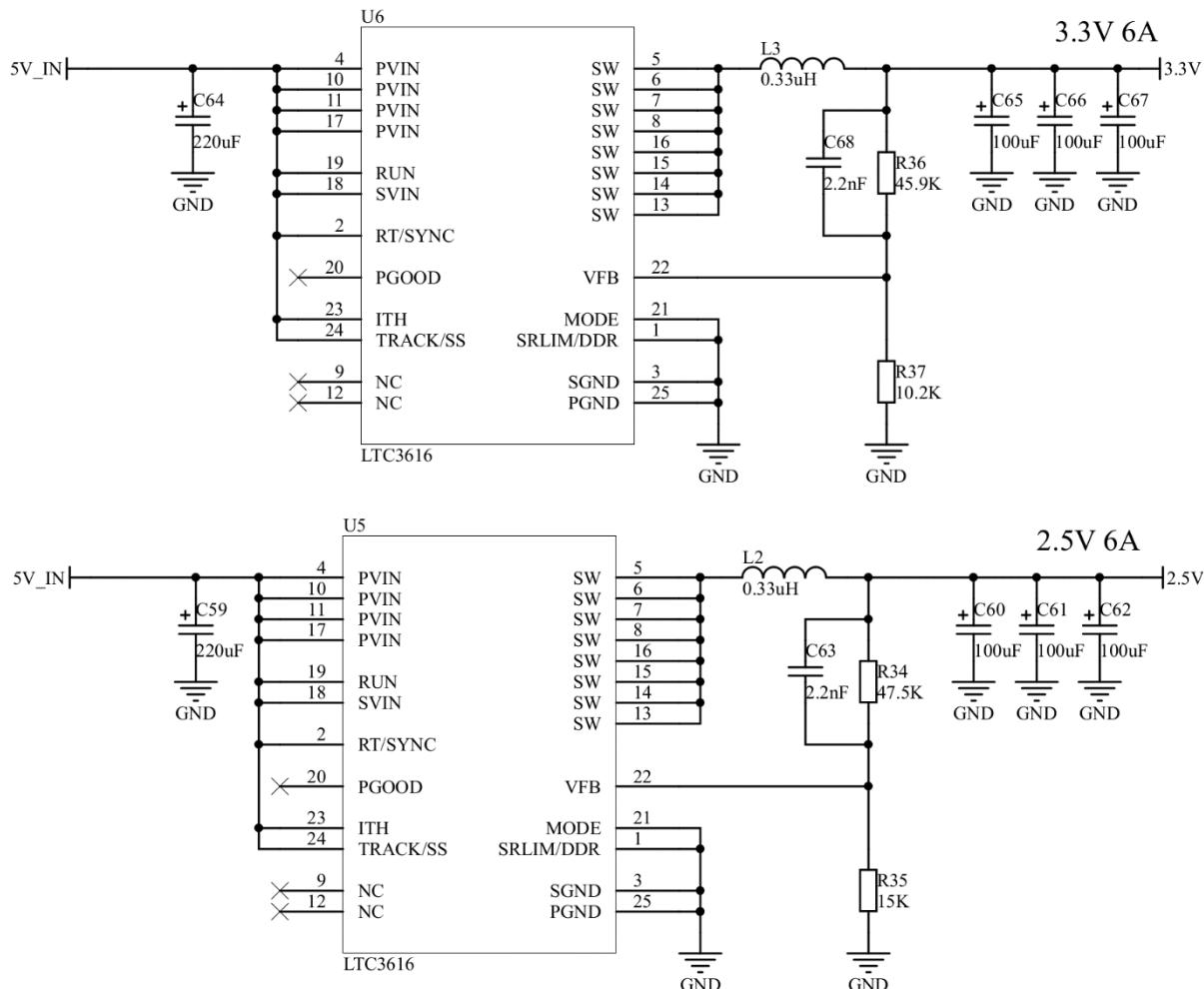


Rys. 27. Schemat stabilizatora impulsowego, obniżającego napięcie do 1.2V

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.

Rysunek 27 przedstawia schemat stabilizatora impulsowego obniżającego napięcie do wartości 1.2V, które używane jest przede wszystkim do zasilania logiki układu FPGA, a także, jako jedno z napięć bufora obrazu. Układ stabilizatora U4 jest w stanie zasilić odbiornik pobierający maksymalnie 8A. Rezystory R29 i R31 tworzą dzielnicę napięcia, który służy do konfiguracji napięcia wyjściowego. Opornik R33 ustawa częstotliwość pracy układu na ok. 2.24MHz [12].



Rys. 28. Schemat stabilizatorów impulsowych, obniżających napięcie do 2.5V i 3.3V

Na rysunku 28 przedstawiono stabilizatory impulsowe, które na wyjściu dają napięcie 3.3V oraz 2.5V. Pierwsza wartość napięcia wykorzystywana jest między innymi do zasilania pamięci SSRAM, kontrolera wejścia/wyjścia układu FPGA. Napięcie 2.5V jest używane w FPGA do zasilania układów PLL. W obu przypadkach wykorzystano te same układy, pozwalające na podłączenie obciążenia pobierającego prąd o natężeniu 6A. Wartości rezystorów R36 i R37 oraz R34 i R35 ustalają napięcie wyjściowe. Układy U5 i U6 pracują z domyślną częstotliwością [13].

4. 2. Projekt i wykonanie obwodu drukowanego modułu SNF-0

W poniższym podrozdziale został opisany proces projektowania obwodu drukowanego modułu SNF-0. Przedstawiono szczegóły techniczne projektu i zastosowane rozwiązania. Dalsze podrozdziały prezentują proces montażu i uruchomienia modułu.

4. 2. 1. 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 rozdziale trzecim.

Porty wejścia/wyjścia, przyciski i kontrolki zostały ułożone na brzegach płytka, 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 wykorzystaniu wysokich częstotliwości. Starano się także, aby ważne połączenia były jak najkrótsze. Odpowiednio zwiększo grubość ścieżek układu zasilania, którymi może płynąć prąd o wysokim natężeniu. Zastosowano opisy na obu stronach płytki PCB, które ułatwiały montaż elementów i stanowią podstawowy opis dostępnych portów i kontrolek. W miejscach, w których nie ma ścieżek i elementów zastosowano wypełnienie obszarem miedzi połączonym do masy, w celu redukcji szumów i ułatwienia dostępu masy do wymaganych miejsc [10].

4. 2. 2. Wykonanie płytki obwodu drukowanego i montaż elementów

Wykonanie płytki obwodu drukowanego zostało zlecone firmie zajmującą 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 przeięciami 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 leveling*).

4. 2. 3. 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 przymontowano 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 przymontowaniu 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 podrozdziale.

4. 3. Projekt i wykonanie przykładowego procesora graficznego

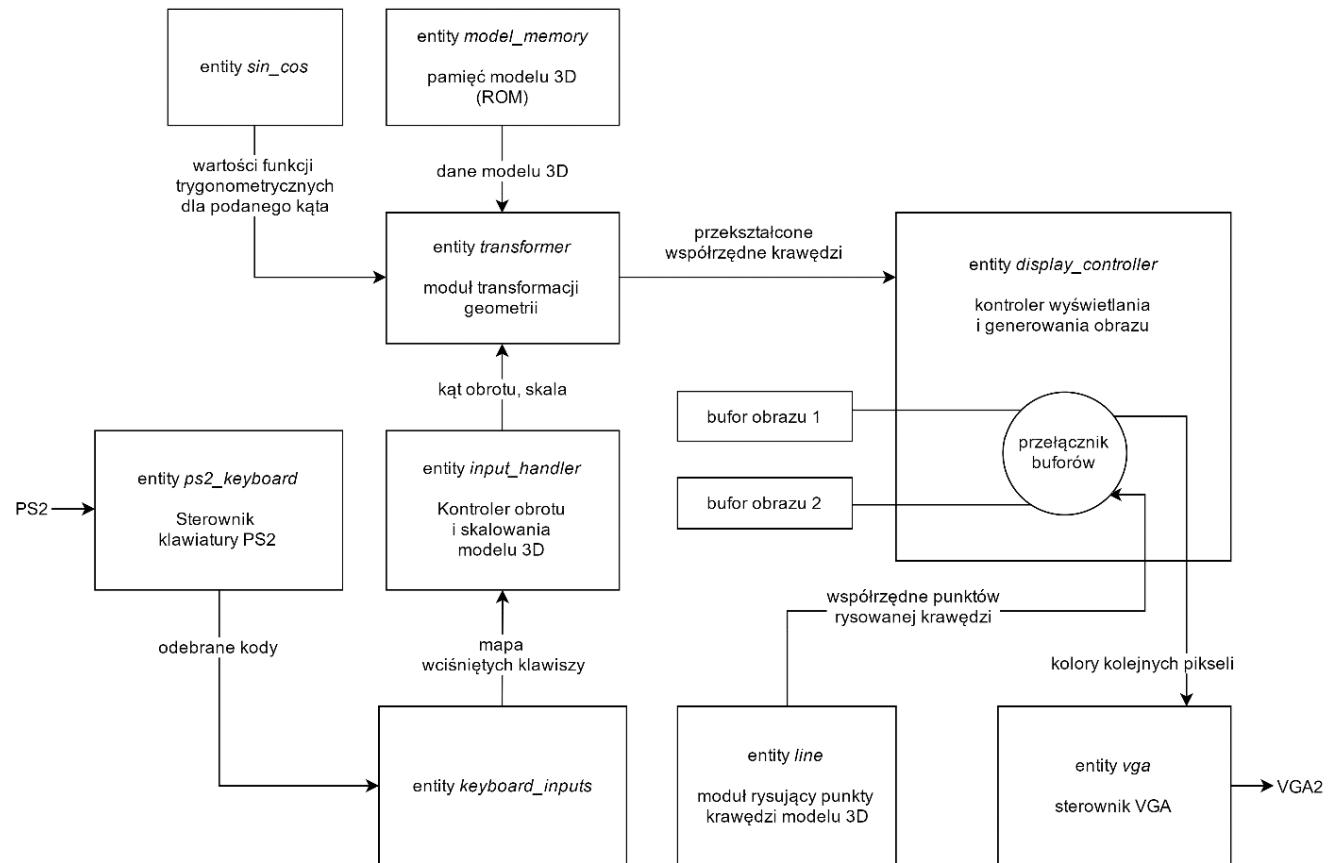
Główym celem projektu inżynierskiego było zbudowanie platformy umożliwiającej tworzenie procesorów zaprojektowanych w językach opisu sprzętu. Aby zaprezentować możliwości modułu SNF-0 wykonano przykładowy specjalizowany procesor graficzny służący do wyświetlania prostej grafiki 3D zbudowanej z linii.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

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

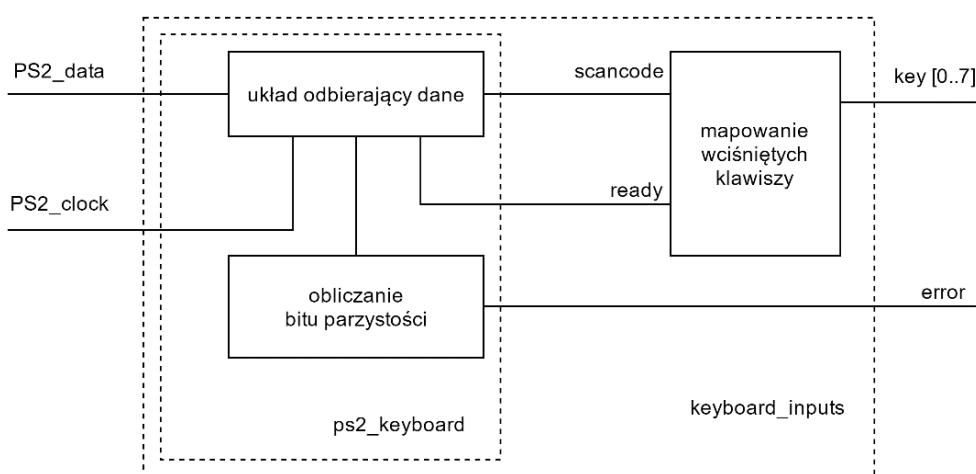


Rys. 29. Schemat blokowy specjalizowanego procesora graficznego

Na rysunku 29 pokazano schemat blokowy procesora graficznego. Poniżej przedstawiono szczegółowy opis jego poszczególnych jednostek.

4. 3. 1. Kontroler 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*.



Rys. 30. Schemat blokowy kontrolera klawiatury PS2

Kontroler klawiatury (jego schemat blokowy przedstawiony jest na rysunku 30) 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. Odebrane dane zapisywane są do rejestrów przesuwnego, taktowanego zegarem *PS2_clock*. Układ odbierający dane posiada także licznik, który odmiera 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.

4. 3. 2. 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.

4. 3. 3. 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 współrzędne 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.

4. 3. 4. Moduł przekształceń geometrycznych

Wstęp

Jednostka projektowa wykonująca transformacje geometryczne jest najważniejszym modelem projektu. Jej funkcją jest obliczanie przekształceń obrotu i skalowania modelu 3D, a także rzutowanie trójwymiarowych współrzędnych na płaszczyznę wyświetlana 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 posiadainstancję pamięci modelu, przechowującą dane określające współrzędne krawędzi tworzących model 3D. Na wyjściu pojawiają się współrzędne x_0, y_0, x_1, y_1 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°]
elseif angle ≥ 180 then
    sin_out ← -sin_lut[angle - 180°]
    cos_out ← -sin_lut[270° - angle]
elseif 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 poniżej tabeli.

Nazwa stanu	Następny stan	Opis
request_sincos_x	get_sincos_x	Przekazanie kąta obrotu wokół osi x do modułu <i>sin_cos</i>
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 <i>sin_cos</i>
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 <i>sin_cos</i>

<i>get_sincos_z</i>	<i>cast_edge_to_32bit</i>	Zapisanie wartość funkcji sinus i cosinus dla wcześniej podanego kąta
<i>cast_edge_to_32bit</i>	<i>apply_x_rot</i>	Skopiowanie danych aktualnej krawędzi do tymczasowej struktury do obliczeń o 32-bitowych wartościach
<i>apply_x_rot</i>	<i>apply_y_rot</i>	Wykonanie obliczeń skutkujących obrótom obiektu wokół osi x
<i>apply_y_rot</i>	<i>apply_z_rot</i>	Wykonanie obliczeń skutkujących obrótom obiektu wokół osi y
<i>apply_z_rot</i>	<i>apply_scale</i>	Wykonanie obliczeń skutkujących obrótom obiektu wokół osi z
<i>apply_scale</i>	<i>inc_line_cnt</i>	Przeskalowanie współrzędnych krawędzi przez podany współczynnik
<i>inc_line_cnt</i>	<i>edge_draw_wait</i> – jeżeli pozostały krawędzie do wyświetlenia <i>next_wait</i> – jeżeli wszystkie krawędzie są narysowane	Zwiększenie licznika krawędzi i ustawienie flagi <i>edge_ready</i> – gdy pozostały jeszcze krawędzie do przekształcenia, wyzerowanie licznika i ustawienie bitu wyjściowego <i>rendered</i> w przeciwnym wypadku
<i>edge_draw_wait</i>	<i>cast_edge_to_32bit</i>	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, wskażywana przez licznik <i>inc_line_cnt</i> .
<i>next_wait</i>	<i>request_sincos_x</i> – jeżeli ustalono flagę <i>update_rot_request</i> <i>cast_edge_to_32bit</i> – w przeciwnym wypadku	Oczekiwanie na sygnał rozpoczęcia kolejnej transformacji krawędzi modelu 3D. Jeżeli ustaliona jest flaga świadcząca o zmianie któregoś z kątów obrotu (<i>update_rot_request</i>), 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 algorytmu przedstawionego poniżej. Wartości $x0$, $y0$, $x1$, $y1$ (w przedziale [-8191, 8191]) oznaczają współrzędne końców linii do narysowania, natomiast struktura *edge* zawiera koординaty 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. Dobrały 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 31). 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 [9].

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

4. 3. 5. Kontroler wyświetletania

Jednostka kontrolera wyświetletania jest kolejnym ważnym modelem 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 zwiększo 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 wyświetlonego 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
<i>start_draw</i>	<i>clear</i>	Aktualny adres bufora zapisu ustawiany jest na 0, w stan wysoki przechodzi flaga <i>start</i> , która informuje o rozpoczęciu generowania sceny jednostce transformującej geometrię, która przygotowuje pierwszą krawędź.
<i>clear</i>	<i>start_transform</i> – po skończeniu czyszczenia bufora	Następuje przypisanie koloru tła do wszystkich pikseli bufora zapisu.
<i>start_transform</i>	<i>transform_wait</i> – jeżeli nie przekształcono jeszcze wszystkich krawędzi <i>frame_rendered</i> – w przeciwnym przypadku	Ustawienie flagi zlecającej rozpoczęcie transformacji kolejnej krawędzi, jeżeli jeszcze nie wszystkie zostały przekształcone.
<i>transform_wait</i>	<i>load</i> – jeżeli scena nie została wyrenderowana i krawędź dostępna jest przekształcona krawędź <i>frame_rendered</i> – jeżeli scena została wyrenderowana	Oczekiwanie na przekształcenie krawędzi przez moduł <i>transformer</i> .
<i>load</i>	<i>init_draw</i>	Skopiowanie przekształconych współrzędnych do rejestrów generatora linii.
<i>init_draw</i>	<i>draw_edge</i>	Rozpoczęcie rysowania linii
<i>draw_edge</i>	<i>start_transform</i> – jeżeli cała linia została wygenerowana	Pobieranie kolejnych współrzędnych pikseli z generatora linii i ich zapis w pamięci obrazu.
<i>frame_rendered</i>	<i>start_draw</i> – po zamianie buforów	Oczekiwanie na zlecenie zamiany buforów od procesu wyświetlającego obraz na monitorze VGA i wykonanie tej akcji.

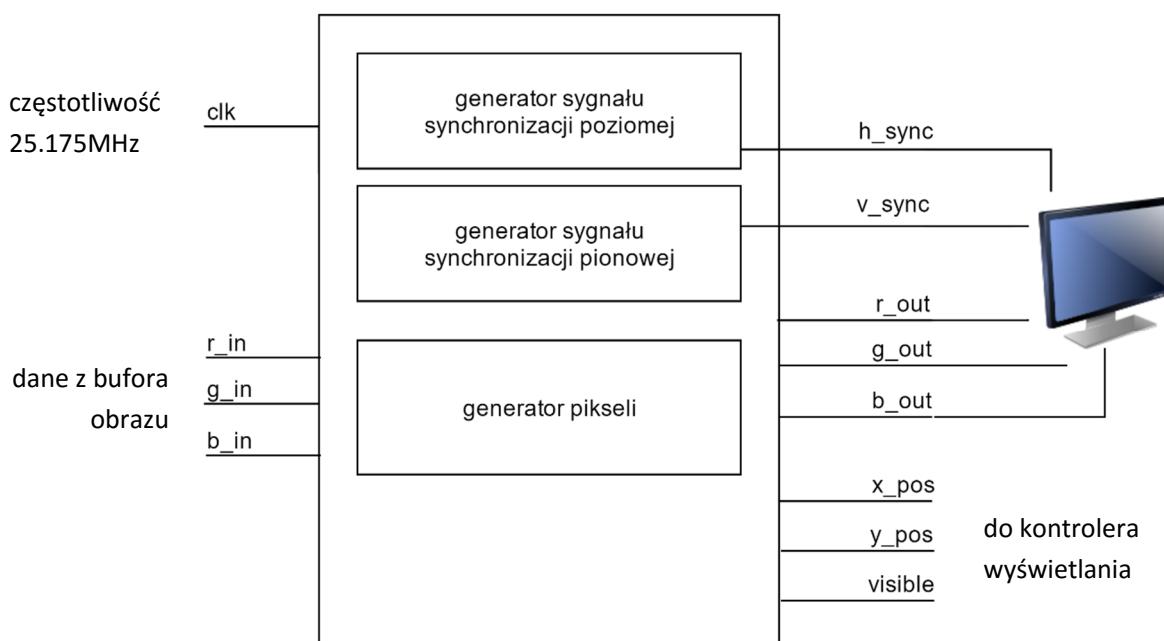
4. 3. 6. Generator odcinków reprezentujących krawędzie obiektu 3D

Do generowania odcinków (krawędzi bryły 3D) 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.

4. 3. 7. Kontroler VGA

Jednostka projektowa kontrolera VGA na wyjściu, oprócz wartości kolorów, generuje sygnały synchronizacji poziomej i pionowej. Dostępne są także rejesty *x_pos* i *y_pos*, które zawierają współrzędne aktualnie wyświetlonego 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 [8]. Podłączenie modułu przedstawiono na rysunku 32.



Rys. 32. Jednostka kontrolera VGA

4. 3. 8. 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 *EditPoly* 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.

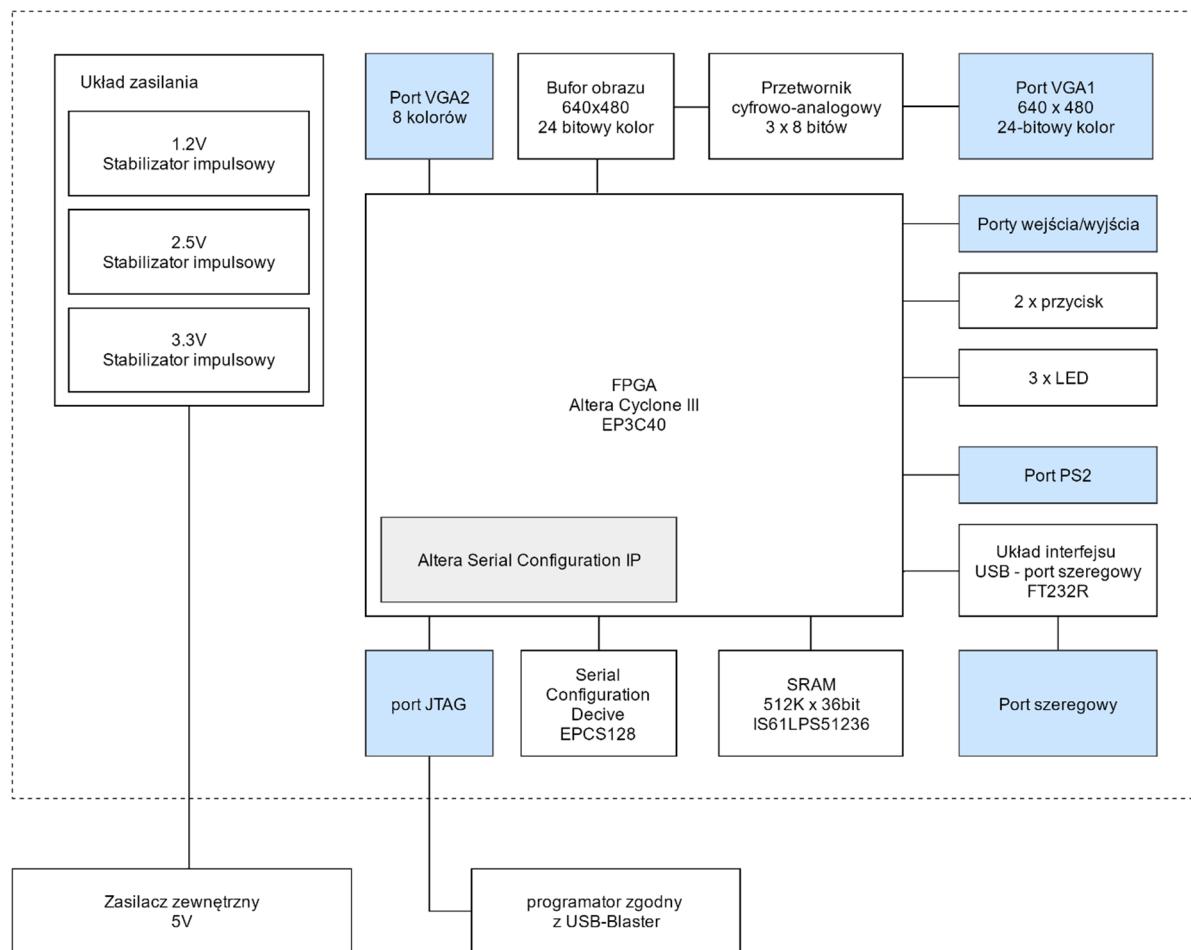
Tak przygotowany plik **.vhd* może być w łatwy sposób użyty w projekcie specjalizowanego procesora graficznego.

5. Prezentacja gotowego produktu

W rozdziale piątym zaprezentowano wynik pracy inżynierskiej: opisano budowę i podstawowe elementy zestawu SNF-0. Przedstawiono także schemat blokowy modułu i wymieniono parametry poszczególnych jego podzespołów. Na koniec opisano podstawowe metody wykorzystania gotowego produktu.

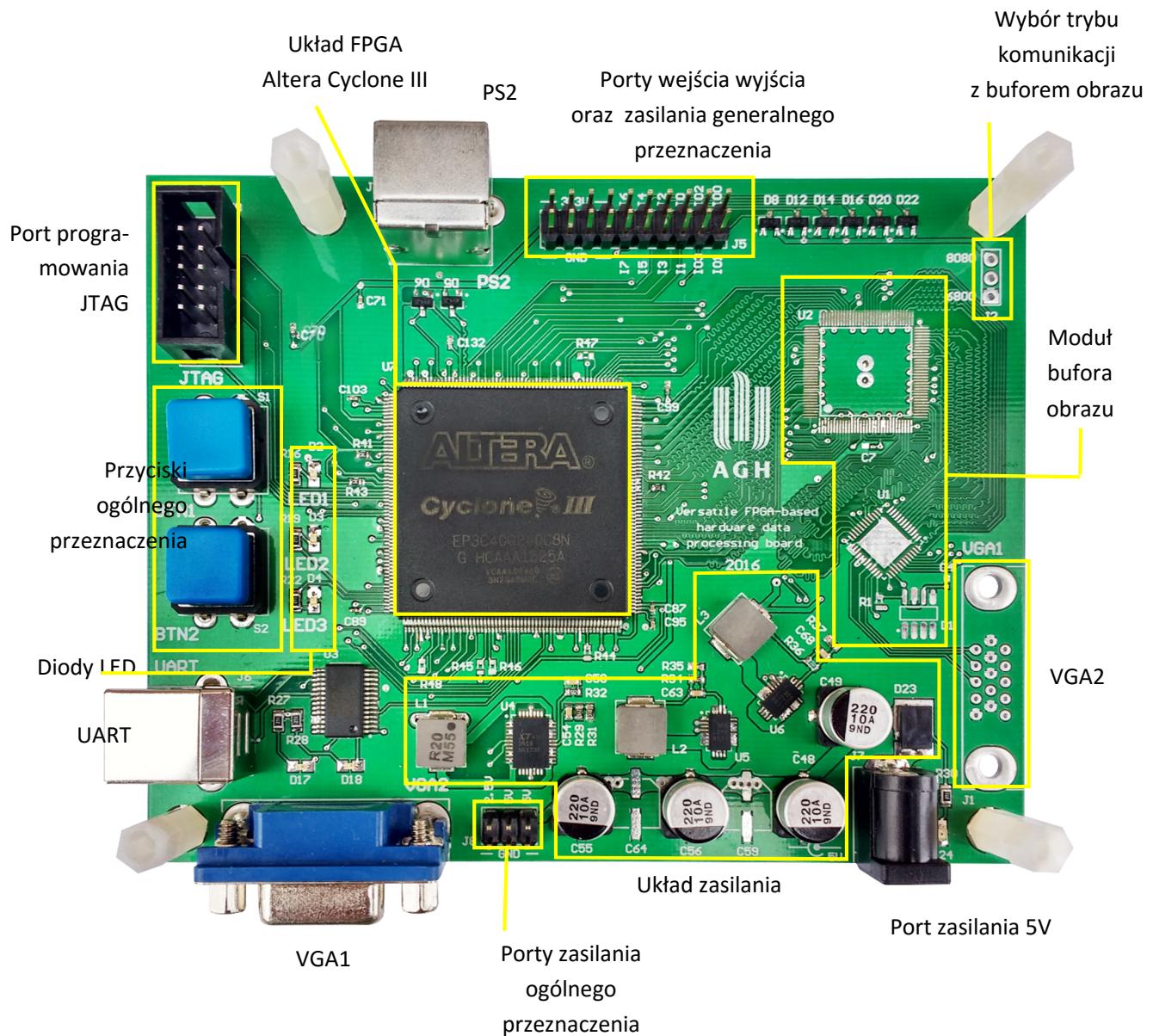
5. 1. Opis elementów płytki PCB modułu SNF-0

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. Na rysunku 33 przedstawiono schemat blokowy modułu.



Rys. 33. Schemat blokowy modułu

Rysunek 34 pokazuje zdjęcie modułu SNF-0 z opisem podstawowych jego elementów składowych. Poniżej opisano dostępne porty i układy.



Rys. 34. Porty i układy modułu uruchomieniowego SNF-0

Dostępne porty:

- VGA1 (8 kolorów, dowolna rozdzielcość)
- VGA2 (24-bitowa głębia kolorów, dostępny bufor obrazu, maksymalna rozdzielcość: 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 wejście 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 EPICS128)
- 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)

5. 2. Szczegółowe parametry podzespołów modułu SNF-0

W podrozdziale 5. 2. opisano podstawowe parametry najważniejszych elementów modułu SNF-0.

Układ FPGA EP3C40

- 39 600 jednostek logicznych
- 1 161 216 bitów pamięci RAM (126 bloków M9K)
- 126 18-bitowych układów mnożących
- 4 układy PLL
- 128 portów wejścia/wyjścia
- Obudowa PQFP240

Moduł programowania

- Programowanie w trybie JTAG
- 128-megabitowa pamięć EEPROM (*Serial Configuration Device*) do przechowywania konfiguracji układu FPGA (zgodna z EPICS128)
- Możliwość programowania układu *Serial Configuration Device* przy użyciu portu JTAG
- Zewnętrzny programator zgodny z Altera USB-Blaster

Pamięć SRAM

- Pojemność: 2MB
- Szerokość magistrali danych: 32 bity + 4 bity parzystości
- *Burst mode* – pozwala na wydajniejsze przesyłanie porcji danych
- Maksymalna częstotliwość pracy: 200MHz

Przyciski i kontrolki LED

- 3 diody LED generalnego przeznaczenia
- 2 przyciski
- Zastosowano prosty filtr do eliminacji efektu odbicia styków

Wyjście VGA1

- 8 bitów na każdy kanał koloru
- Używa bufora obrazu mogącego pomieścić maksymalnie obraz o rozdzielczości 864 x 480 i 24-bitowej głębi koloru
- Bufor może komunikować się z układem FPGA przy użyciu dwóch trybów 6800 i 8080
- Bufor posiada także dodatkowe opcje zmiany parametrów obrazu, takie jak kontrola jasności, kontrastu, nasycenia i obrotu
- Wyjście bufora połączone do 3 kanałowego, 8 bitowego przetwornika cyfrowo-analogowego

Wyjście VGA2

- 1 bit na każdy kanał koloru
- Sterowany bezpośrednio przez układ FPGA (używając diod zabezpieczających)
- 8 kolorów możliwych do uzyskania
- Możliwość użycia dowolnej rozdzielczości

Porty szeregowe

- Port szeregowy zgodny ze standardem RS-232
- Moduł SNF-0 zawiera wbudowany konwerter port szeregowy – USB, wykorzystujący układ *FTDI FT232R*, co pozwala na łatwe podłączenie modułu do nowszych komputerów nieposiadających złącza standardu RS-232
- Port PS2, pozwalający na podłączenie klawiatury lub myszki

Porty wejścia/wyjścia ogólnego przeznaczenia

- 4 porty wejścia/wyjścia
- 8 portów wejściowych
- 4 wyprowadzenia napięcia zasilania 3.3V do zasilenia zewnętrznych modułów
- 1 wyprowadzenie napięcia 2.5V
- 2 wyprowadzenia napięcia 5V

Układ zasilania

- Napięcie wejściowe: 5V
- 3 stabilizatory impulsowe
- Stabilizatory 2.5V i 3.3V, 6A
- Stabilizator 1.2V, 8A

5. 3. Instrukcja uruchomienia i wykorzystania modułu SNF-0

W podrozdziale 5. 3. opisano peryferia sprzętowe potrzebne do uruchomienia modułu SNF-0 oraz wymieniono wymagane oprogramowanie. Podręczniki, dotyczące instalacji sterowników i przygotowania projektu w oprogramowaniu *Quartus II* do uruchomienia na module SNF-0, znajdują się w dokumentacji użytkownika w rozdziale 3.

5. 3. 1. Peryferia sprzętowe

- Komputer z systemem Microsoft Windows
- Programator zgodny z USB-Blaster
- Monitor ze złączem VGA
- Klawiatura lub myszka z portem PS2
- Kabel USB-B
- Zasilacz sieciowy 5V

Oprócz modułu SNF-0 w zestawie dostępny jest zasilacz sieciowy o napięciu wyjściowym 5V, który ze względów bezpieczeństwa, należy podłączyć do modułu przed innymi peryferiami.

Aby w pełni wykorzystać możliwości modułu SNF-0 potrzebny jest monitor ze złączem VGA, obsługujący rozdzielcość minimum 640 x 480 pikseli i potrafiący wyświetlać kolorowy obraz. Złącze monitora należy podpiąć do wyjścia VGA2 modułu uruchomieniowego.

Opcjonalne jest podłączenie modułu do komputera za pomocą kabla USB-B, które umożliwia komunikację z układem FPGA poprzez port szeregowy. Wymagane jest posiadanie odpowiednich sterowników oraz programu do komunikacji przez port szeregowy. Instalacja odpowiedniego oprogramowania zostanie omówiona w następnym podrozdziale.

Do modułu SNF-0 można podpiąć także klawiaturę lub myszkę z kablem PS2.

Przed przystąpieniem do programowania układu FPGA, należy przygotować programator zgodny z *Altera USB-Blaster* (dołączony do zestawu). Urządzenie należy podłączyć do portu JTAG na płytce modułu SNF-0 oraz do portu USB komputera. Instalacja sterowników i programowania układu FPGA zostanie opisana w rozdziale trzecim dokumentacji użytkownika.

W repozytorium projektu, w folderze *HDL/SNF0_empty* znajduje się skonfigurowany pusty projekt programu *Altera Quartus II 13.1*, który można wykorzystać, jako bazę do rozbudowy.

5. 3. 2. Zalecane oprogramowanie

Do odpowiedniego wykorzystania zestawu, potrzebne będzie następujące oprogramowanie:

- System Microsoft Windows XP lub nowszy
- Altera Quartus II z bibliotekami Cyclone III (do wersji 13.1)
- Sterownik Altera USB-Blaster (dołączony do oprogramowania Quartus II)
- Sterownik interfejsu USB FTDI
- Autodesk 3DS Max 2010 lub nowszy (opcjonalnie)
- Terminal wspierający obsługę portu szeregowego (np. Realterm) (opcjonalnie)

5. 3. 3. Wspierane wersje oprogramowania Quartus II

Do programowania i przygotowywania konfiguracji układu FPGA może zostać wykorzystane oprogramowanie *Quartus II* firmy *Altera* w wersji *Web Edition*, jak również w *Subscription Edition*.

Należy także zainstalować biblioteki dodające wsparcie dla układów FPGA *Cyclone III*. Maksymalna wersja programu *Quartus II* posiadająca takie wsparcie to 13.1.

5. 4. Opis wykorzystania procesora graficznego

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

Aby uruchomić ten przykład, należy przygotować monitor VGA i klawiaturę PS2, zgodnie z opisem z podrozdziału 5. 3. 1.

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. Szczegółowy opis tych jednostek projektowych znajduje się w dokumentacji technicznej projektu inżynierskiego.

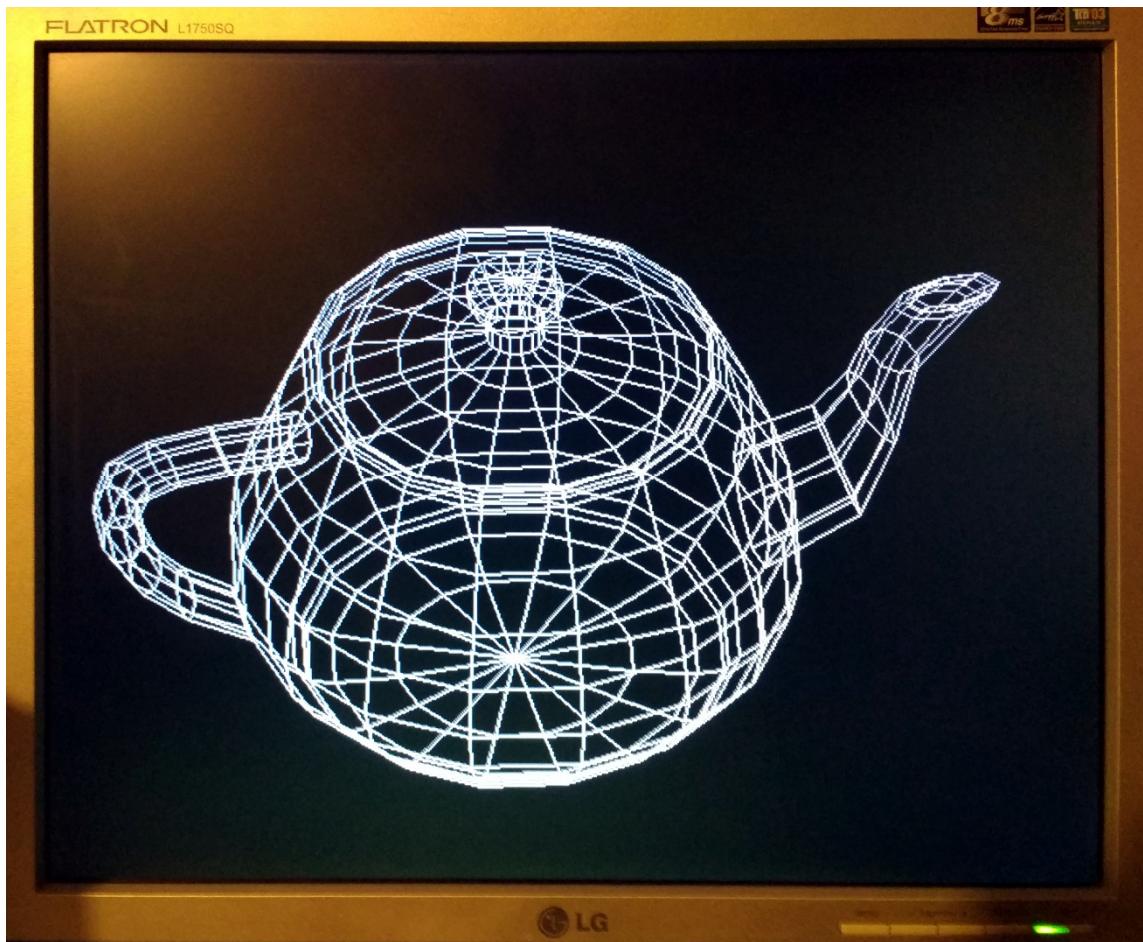
5. 4. 1. Kompilacja i uruchomienie projektu na module SNF-0

Projekt jest dostępny w katalogu *HDL/SNF0_3D_processor* w repozytorium projektu inżynierskiego dostępnego pod adresem:

https://github.com/papciakk/FPGA_Board (dostęp 2.11.2016)

Po zimportowaniu projektu do programu Altera Quartus II, należy go skompilować za pomocą opcji *Start compilation* w menu *Processing*. Po udanej komplikacji projekt powinien być gotowy do zaprogramowania układu FPGA, zgodnie z instrukcją podaną w podrozdziale 3.5 dokumentacji użytkownika.

Jeżeli do modułu został podpięty monitor i klawiatura PS2, możliwe będzie przetestowanie działania programu.



Rys. 35. Działanie przykładowego specjalizowanego procesora graficznego

5. 4. 2. Sterowanie przy pomocy klawiatury PS2

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

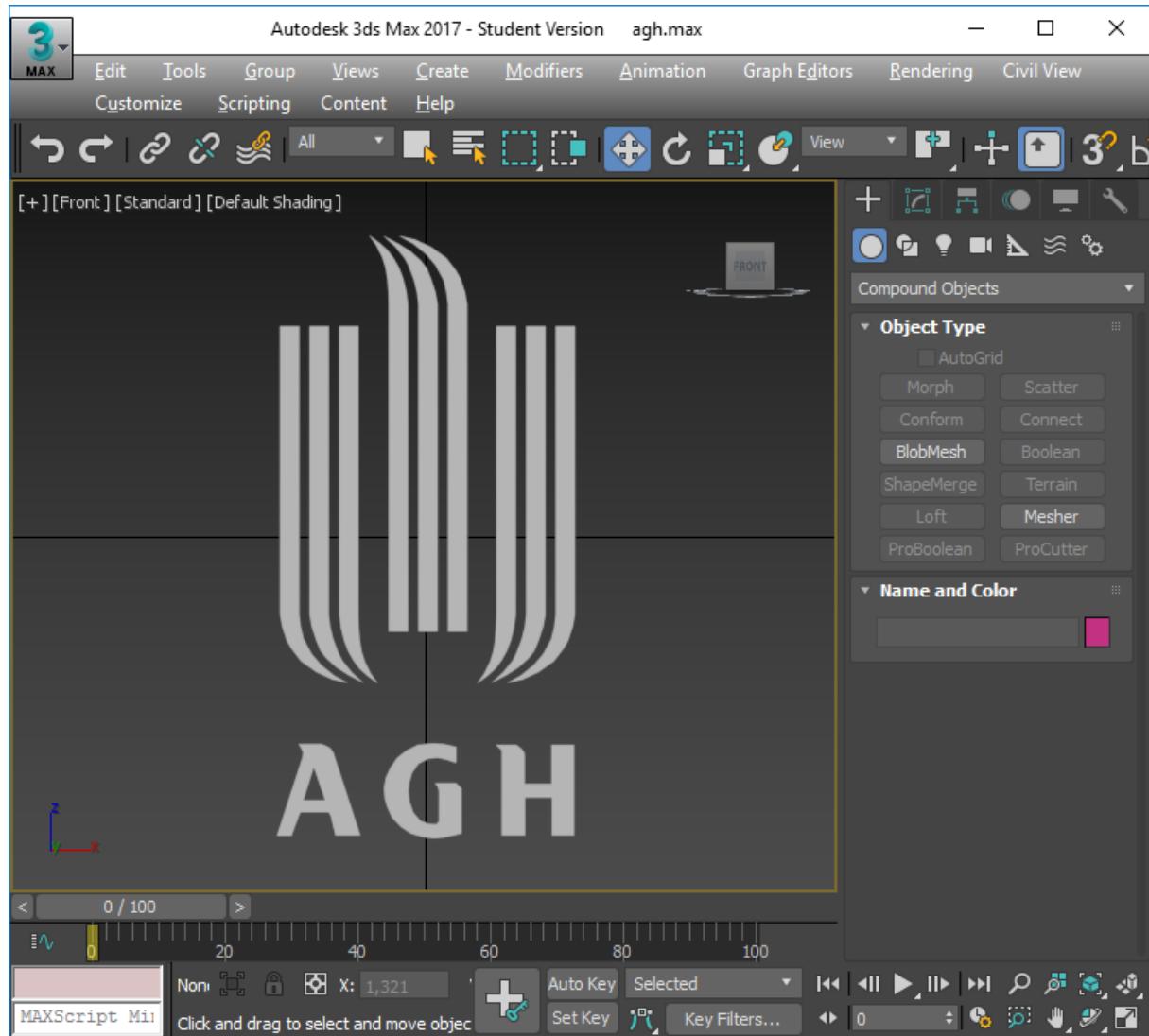
- W – zwiększenie kąta obrotu względem osi Y
- S – zmniejszenie kąta obrotu względem osi Y
- A – zwiększenie kąta obrotu względem osi X
- D – zmniejszenie kąta obrotu względem osi X
- Q – zwiększenie kąta obrotu względem osi Z
- E – zmniejszenie kąta obrotu względem osi Z
- Z – zwiększenie skali obiektu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

5. 5. Instrukcja przygotowania własnych modeli do wyświetlenia przy pomocy procesora graficznego

W podrozdziale 5. 5. podano instrukcję przygotowania modelu w programie Autodesk 3DS Max do wyświetlenia przy pomocy specjalizowanego procesora graficznego na module SNF-0.

5. 5. 1. Przygotowanie modelu w programie Autodesk 3DS Max

Do wykonania poniższych czynności potrzebne będzie oprogramowanie Autodesk 3DS Max, w wersji co najmniej 2010.



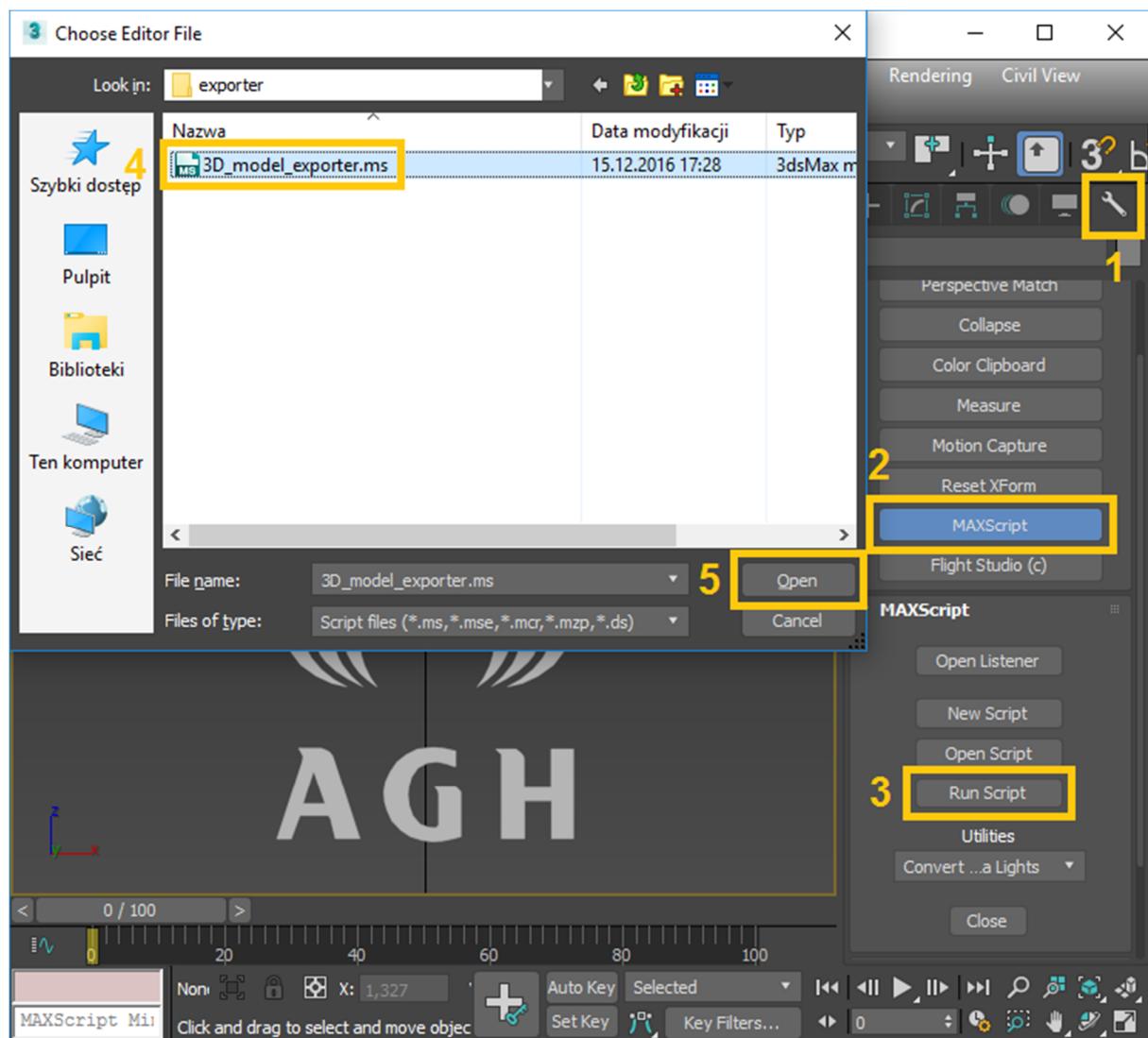
Rys. 36. Scena w programie 3DS Max. Ilość krawędzi jest mniejsza niż 5500.

Pierwszym krokiem jest przygotowanie modelu 3D, który chcemy wyświetlić. W razie potrzeby należy zaimportować model przy pomocy odpowiedniego plug-inu, obsługującego wymagany format. Jeżeli model składa się z więcej niż jednego obiektu, należy je ze sobą połączyć (np. przy narzędzia *collapse*

w bocznym menu *utilities*). Ważne jest, aby model posiadał nie więcej niż **5500 krawędzi**. Ograniczenie to jest związane z pojemnością pamięci RAM układu FPGA. Rysunek 36 pokazuje przykładową scenę przygotowaną w programie *3DS Max*.

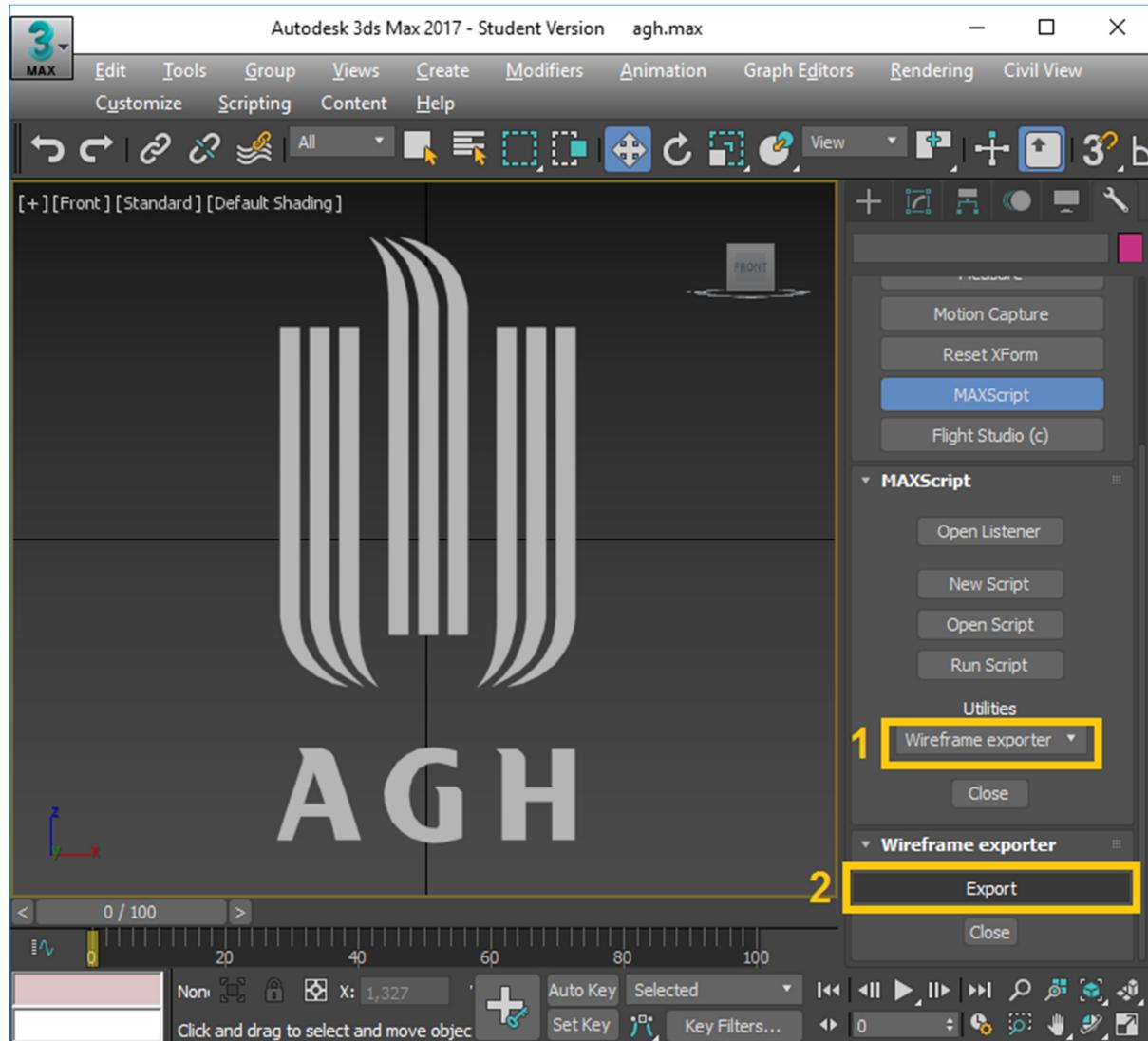
5. 5. 2. Eksport modelu przy użyciu skryptu

Kiedy scena do wyświetlenia na procesorze graficznym jest gotowa, należy wczytać skrypt eksportu *3D_model_exporter.ms*. Znajduje się on w repozytorium projektu w folderze *software/exporter*. Aby to zrobić należy najpierw otworzyć panel języka skryptowego *MAXScript*. Należy kliknąć w opcję *Utilities* w menu bocznym i wybrać opcję *MAXScript*. Następnie w otworzonym panelu trzeba kliknąć w przycisk *Run script* i wybrać eksporter *3D_model_exporter.ms*. Kolejne kroki całej operacji zostały przedstawione na rysunku 37.



Rys. 37. Wczytywanie skryptu eksportu modelu w programie 3DS Max

Po właściwym wczytaniu skryptu, należy wybrać go z listy rozwijanej *Utilities* (figuruje w niej jako *Wireframe exporter*). Pojawia się przycisk *Export scene objects*. Po jego naciśnięciu, wyświetli się dialog wyboru ścieżki zapisu pliku z modelem 3D. Zostaną do niego wyeksportowane modele znajdujące się w aktualnej scenie projektu 3DS Max. Proces przedstawiono na rysunku 38.



Rys. 38. Eksport modelu w programie 3DS Max do wykorzystania na procesorze graficznym

5. 5. 3. Przygotowanie do wyświetlania modelu na module SNF-0

Po wyeksportowaniu modelu z programu 3DS Max przy użyciu dołączonego skryptu, należy skopiować go do folderu projektu procesora graficznego, do katalogu *src* i nazwać *model.vhd*. Poniżej przedstawiono fragment wygenerowanego pliku *vhd*, który zawiera dane modelu 3D zapisane z postaci szesnastkowej.

Następnie należy otworzyć projekt i go skompilować. Po zaprogramowaniu układu FPGA, na monitorze powinna pojawić się wyeksportowana scena.

```
use work.typedefs.all;

package model is
    constant MODEL_EDGES_NUMBER : integer := 656;
    constant MODEL_DATA : mesh_data(655 downto 0) := (
        X"0DBF01F4F7C70D9601F4F67C",
        X"0D9601F4F67C0D96FE0CF67C",
        X"0D96FE0CF67C0DBFFE0CF7C7",
        X"0DBFFE0CF7C70DBF01F4F7C7",
        ...
    );
```



Rys. 39. Moduł SNF-0 wyświetlający wyeksportowany model 3D

Rysunek 39 przedstawia wynik działania programu z wyeksportowanym modelem 3D.

6. Analiza wykonanych prac

Rozdział szósty obejmuje analizę realizacji założeń projektowych. Przedstawiono także plany rozwoju projektu.

6. 1. 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.

6. 2. 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 paletie kolorów.

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

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.

7. 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](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] Waveshare. *USB Blaster V2 Datasheet*
http://www.waveshare.com/wiki/USB_Blaสเตr_V2 (dostęp 15.12.2016)
- [8] Pong P. Chu. *FPGA Prototyping by VHDL Examples*. John Wiley & Sons, Inc., 2008
- [9] OpenGL Transformations
<https://open.gl/transformations> (dostęp: 10.11.2016)
- [10] Altium. *PCB Routing, 2014*
<http://techdocs.altium.com/display/ADOH/PCB+Routing> (dostęp 10.11.2016)
- [11] Altera. *Serial Configuration Devices Datasheet, 2014*
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cfg/cyc_c51014.pdf (dostęp 10.11.2016)
- [12] Linear Technology. *LTC3418 Datasheet*
<http://cds.linear.com/docs/en/datasheet/3418fb.pdf> (dostęp 15.12.2016)
- [13] Linear Technology, *LTC3616 Datasheet*
<http://cds.linear.com/docs/en/datasheet/3616fc.pdf> (dostęp 15.12.2016)

8. Spis rysunków

Rys. 1. Schemat koncepcyjny jednostki logicznej opartej o tablicę LUT	5
Rys. 2. Struktura układu FPGA rodziny Cyclone II i opis podstawowych elementów	6
Rys. 3. Element logiczny układu Cyclone II w trybie normalnym	7
Rys. 4. Element logiczny układu Cyclone II w trybie arytmetycznym	8
Rys. 5. Układ FPGA Cyclone III na płytce modułu SNF-0	9
Rys. 6. Początkowa wersja układu zasilania	15
Rys. 7. Prototyp obwodu drukowanego modułu SNF-0	17
Rys. 8. Projekt obwodu drukowanego widziany z góry	19
Rys. 9. Projekt obwodu drukowanego widziany z dołu	19
Rys. 10. Wizualizacja modułu SNF-0	20
Rys. 11. Moduł SNF-0 podczas montażu	20
Rys. 12. Gotowy moduł SNF-0 (widok od góry)	21
Rys. 13. Gotowy moduł SNF-0 (widok od dołu)	21
Rys. 14. Prototypowy program o funkcjonalności projektowanego procesora graficznego	23
Rys. 15. Podział i połączenie dokumentów schematu modułu SNF-0	24
Rys. 16. Schemat podłączenia portów wejścia/wyjścia układu FPGA	25
Rys. 17. Schemat układu programowania i taktowania modułu SNF-0	26
Rys. 18. Schemat podłączenia pamięci SSRAM do układu FPGA	27
Rys. 19. Schemat interfejsu portu szeregowego	28
Rys. 20. Schemat podłączenia portu PS2	29
Rys. 21. Schemat podłączenia diod LED i przycisków do układu FPGA	29
Rys. 22. Schemat portów wejścia/wyjścia ogólnego przeznaczenia	30
Rys. 23. Schemat wyjścia VGA2	31
Rys. 24. Schemat połączeń układu bufora obrazu i jego otoczenia	32
Rys. 25. Schemat wyjścia VGA1 i przetwornika cyfrowo-analogowego	33
Rys. 26. Schemat złącza zasilania i elementów peryferyjnych	34
Rys. 27. Schemat stabilizatora impulsowego, obniżającego napięcie do 1.2V	34
Rys. 28. Schemat stabilizatorów impulsowych, obniżających napięcie do 2.5V i 3.3V	35
Rys. 29. Schemat blokowy specjalizowanego procesora graficznego	38
Rys. 30. Schemat blokowy kontrolera klawiatury PS2	39
Rys. 32. (a) Macierze obrotu dla poszczególnych współrzędnych, (b) macierz skalowania, (c) złożenie przekształceń i obliczenie wektora wynikowego	44
Rys. 32. Jednostka kontrolera VGA	46
Rys. 33. Schemat blokowy modułu	48
Rys. 34. Porty i układy modułu uruchomieniowego SNF-0	49
Rys. 35. Działanie przykładowego specjalizowanego procesora graficznego	56
Rys. 36. Scena w programie 3DS Max. Ilość krawędzi jest mniejsza niż 5500	57
Rys. 37. Wczytywanie skryptu eksportu modelu w programie 3DS Max	58
Rys. 38. Eksport modelu w programie 3DS Max do wykorzystania na procesorze graficznym	59
Rys. 39. Moduł SNF-0 wyświetlający wyeksportowany model 3D	60