POLITECHNIKA POZNAŃSKA
WYDZIAŁ ELEKTRONIKI I TELEKOMUNIKACJI
KATEDRA TELEKOMUNIKACJI MULTIMEDIALNEJ I MIKROELEKTRONIKI

STRESZCZENIE DYSERTACJI

Realizacja zaawansowanych kodeków wizyjnych z wykorzystaniem sieci w układach FPGA

Autor:
Marta STĘPNIEWSKA

Promotor:
Prof. dr hab. inż. Marek DOMAŃSKI

Poznań 2012

Spis treści

Rozdział 1

Wstęp

1.1 Stan obecny

Skompresowane sekwencje wizyjne wykorzystywane są w coraz większej liczbie urządzeń, nie tylko telewizorach i komputerach, ale również w urządzeniach mobilnych, takich jak smartfony i tablety. Zastosowania te nakładają szczególne wymagania na jakość bezprzewodowego łącza danych oraz na wydajność stosowanych rozwiązań kompresji. Rozprawa dotyczy najnowszych technik kompresji sekwencji wizyjnych i odpowiadających im norm, takich jak Advanced Video Coding (AVC) [1], która jest powszechnie stosowana na przykład w systemach telewizyjnych. Istnieją również inne ostatnio zaproponowane normy takie jak VC-1 [3] i Audio Video Standard (AVS) [2]). Równolegle do nich rozwijana jest nowa technika kompresji High Efficiency Video Coding (HEVC) [5, 15], która jednak nie jest jeszcze zaakceptowanym standardem kompresji.

Wśród innych trendów nakładających wysokie wymagania urządzeniom pamięci masowej oraz możliwościom kanału (tj. przepustowości) jest przesyłanie sekwencji wizyjnych stereoskopowych i wielowidokowych. W takim przypadku wymagane jest przesyłanie znacznie większej ilości danych, niż w przypadku tradycyjnych systemów wykorzystujących sekwencje jednowidokowe [6]. Aby spełnić narzucone wymagania kodek wizyjny musi być zrównoleglony, co oznacza, że niektóre jego części muszą być powielone, aby rozszerzyć możliwości obliczeniowe kodeka. Inne podejście, przedstawione w [13], zakłada zwielokrotnienie całego kodeka, aby uzyskać wymaganą moc obliczeniową. Implementacje sprzętowe wymienionych technik przetwarzania danych wizyjnych składają się z wielu modułów, które realizują algo-

rytmy o wysokim zapotrzebowaniu na przepustowość na wejściu i wyjściu poszczególnych modułów. Techniki te charakteryzują się również zmiennością ścieżki przetwarzanych danych (tj., w różnych trybach pracy wykorzystywane są różnie połączone ze sobą grupy modułów). Rosnąca liczba elementów i ich zapotrzebowanie na przepustowość powoduje problemy połączeniowe ze względu na wymaganą liczbę połączeń i zapewnienie przez nie właściwej przepustowości. Duża liczba połączeń powoduje trudności w uzyskaniu prawidłowego trasowania i rozmieszczenia modułów w wynikowym układzie. W przypadku dużej liczby połączeń pomiędzy modułami w urządzeniach FPGA, można próbować trasować połączenia przez tablice przeglądowe (LUT), co jednak powoduje dodatkowe zużycie zasobów sprzętowych.

Odpowiedzią na przedstawione problemy są sieci w układzie (Network-on-Chip - NoC), jak przedstawiono to w rozdziale 1.3 dysertacji. Jednak, jak wskazano w rozdziale 3.2 rozprawy, zastosowania przetwarzania sekwencji wizyjnych są uważane za stosunkowo małe, aby były zrealizowane z wykorzystaniem sieci. Wdrożenie rozwiązań sieci w układzie niesie za sobą koszty sprzętu, które w przypadku aplikacji przetwarzających sygnał wizyjny są znaczące. Niemniej jednak rosnąca złożoność technik przetwarzania danych wizyjnych wymaga nowej architektury połączeń, które oferują zadowalającą przepustowość i skalowalność przy niskich kosztach sprzętu [7, 12].

Innym problemem jest modelowanie aplikacji przetwarzających dane wizyjne (rozdział 5.2 dysertacji), ponieważ mają one skomplikowany model ruchu, charakteryzujący się wieloma typami danych, co znajduje odzwierciedlenie w wielu trybach kodowania i dekodowania. Do wykonywania symulacji programowej takich systemów stosuje się model aplikacji sprzętowej. Sposób kodowania i dekodowania jest szczegółowo opisany w rozdziale 3.1 rozprawy. Istnieją dwie ścieżki kodowania lub dekodowania, co generuje dwa różne wzorce danych. Oznacza to, że zastosowanie uproszczonego opisu modułów, jak przedstawiono to w [10] nie jest możliwe i pojawia się potrzeba zastosowania w symulatorze odpowiedniego mechanizmu zapewniającego taką możliwość. Chociaż proces symulacji programowej jest mniej czasochłonny niż symulacja sprzętowa, to mimo wszystko wymaga czasu na wykonywanie powtarzanych eksperymentów, aby uzyskać statystycznie wiarygodne wyniki [16]. W niektórych przypadkach, zastosowanie symulacji programowej jest przydatne do obserwacji sposobu wpływu zmian w strukturze połączeń między modułami na ogólną wydajność systemu oraz oceny wielkości tego wpływu. W takiej sytuacji niezbędne staje się zapewnienie odpowiedniego narzędzia, wykorzystującego proste obliczenia dla oszacowania wyników symula-

cji. Ponadto, takie narzędzie może być również zastosowane do analizy różnego rozmieszczenia modułów przetwarzających PE w strukturze układu i wcześniejsze określenie możliwości wynikowej aplikacji.

1.2 Cel i teza pracy

Praca skupia się na przyjęciu sieci NoC jako architektury połączeniowej dla kodeków wizyjnych, jako oferującej dobrą skalowalność i efektywne wykorzystanie pasma transmisji. Nawet pomimo tego, że powstaje dodatkowy koszt w postaci dodatkowego zużycia zasobów sprzętowych. Zaproponowana sieć musi zapewnić równowagę pomiędzy oferowaną funkcjonalnością a wymaganym na jej realizację zużyciem zasobów.

Celem pracy jest rozważenie problemu definiowania architektury połączeń nadającej się dla kodeków, w szczególności dla techniki kompresji AVC. Projektowanie infrastruktury połączeniowej wymaga wstępnej analizy zapotrzebowania na moduły przetwarzające. Analiza taka może być przeprowadzona z wykorzystaniem symulacji, która zawiera model rzeczywistej aplikacji i specyfikację ruchu dla jej modułów. Jak już wcześniej wspomniano, aplikacje przetwarzające dane wizyjne, a zwłaszcza kodeki charakteryzują się skomplikowanym modelem ruchu, który musi odzwierciedlać różne tryby kodowania. Wymaga to zaproponowania opisu modelu ruchu, który obejmowałby te zagadnienia.

Rozprawa ma również na celu zbadanie możliwości przybliżenia wyników symulacji z wykorzystaniem prostych obliczeń, wykorzystujących na przykład model kolejkowy. Takie obliczenia mogłyby znacznie skrócić czas potrzebny na zgrubne porównanie wyników różnych symulacji. Przybliżenie wyniku symulacji pozwoli na wcześniejszą reorganizację modułów w infrastrukturze komunikacyjnej oraz oszacowanie wpływu różnych zmian projektowanej architektury na ogólną wydajność aplikacji bez potrzeby przeprowadzania symulacji. Model taki powinien również pozwalać na estymację opóźnienia wprowadzonego przez każdy z modułów umieszczonych na ścieżce przetwarzanych danych.

Teza rozprawy jest sformułowana w następujący sposób:

Dla sprzętowych zaawansowanych kodeków sekwencji wizyjnych sieci w układzie stanowią konkurencyjne rozwiązanie w stosunku do innych struktur połączeń realizowanych w układzie scalonym. Architektura połączeń tworzonych z wykorzystaniem sieci w układzie może być w

łatwy sposób oceniana za pomocą prostych narzędzi symulacyjnych a także obliczeń analitycznych.

Autorka rozprawy pokazuje korzyści z wdrożenia kodeka z wykorzystaniem sieci i omawia koszty takiego rozwiązania. Podstawą analizy jest fizyczna implementacja kodeka AVC zrealizowanego w Katedrze Telekomunikacji Multimedialnej i Mikroelektroniki. Autorka aktywnie uczestniczyła w jego projektowaniu, wdrażaniu i debugowaniu.

W dalszej części rozprawy autorka pokazuje też sposoby modelowania aplikacji sprzętowych i proponuje nową metodę opisu obejmującą specyfikę ruchu w dekoderze sprzętowym sekwencji wizyjnych. Metoda ta zrealizowana w narzędziu symulacyjnym i sprawdzona pod względem dokładności wyników. Dokładność będzie oceniana przez porównanie wyników symulacji do rzeczywistych wyników wydajności uzyskanych z realizacji sprzętowej.

Autorka pokazuje także, że wyniki symulacyjne mogą być zgrubnie przybliżone za pomocą prostych obliczeń, składających się na model kolejkowy. Możliwość zastosowanie takiego modelowania dla dekodera sekwencji wizyjnych oceniona jest na podstawie analizy dokładności otrzymanych wyników. Analiza dokładności symulacji przeprowadzana jest zgodnie z [17]. Autorzy przedstawiają w rozprawie także zbiór praktyk badawczych pozwalających na uzyskanie powtarzalnych wyników w dziedzinie przetwarzania sygnału.

Wymienione metody modelowania kodeków sekwencji wizyjnych powinny być pomocne w czasie projektowania architektury połączeń również dla innych aplikacji, w szczególności w dziedzinie przetwarzania sekwencji wizyjnych, które charakteryzują się modelem ruchu podobnym do ruchu w kodeku. Zaproponowany model symulacyjny może być stosowany do budowania wszelkich symulacji programowych, których celem jest oszacowanie wyników wydajności sprzętu. Ponadto, zastosowanie modelu kolejkowego do uzyskania przybliżonych wyników symulacjyjnych może być zastsowane do w oceny każdej realizacji sprzętowej o podobnej charakterystyce ruchu. Chociaż rozprawa omawia użycie konkretnego modelu kolejki, to w przypadku innego scenariusza ruchu, można wykorzystać inny model kolejkowy. Autorka pokazuje korzyści z wykorzystania takiego modelowania.

1.3 Przegląd pracy

Rozprawa została podzielona na tematyczne rozdziały, których podsumowanie zaprezentowano poniżej.

Rozdział II zawiera przegląd rozwiązań dotyczących połączeń w układzie, stosowanych obecnie w rozwiązaniach SoC, przedstawienie ich wydajności i możliwości zastosowań.

W rozdziale III została przedstawiona technika kompresji AVC, i jej wymagania dla realizacji sprzętowych.

Rozdział IV zawiera propozycje sprzętowej realizacji kodeka AVC opartej na sieci w układzie (NoC). Przedstawiona realizacja jest wynikiem prac zespołu, której członkiem była autorka. Rozdział omawia oryginalną konstrukcję sieci. Zawarto w nim również omówienie zalet i wad implementacji kodeka z użyciem zaproponowanej sieci.

W rozdziale V zostało zawarte omówienie problemu symulacji kodeków oraz została przedstawiona autorska propozycja dotycząca modelowania takich aplikacji. Opisano również własną propozycję symulatora. Dokładność symulacji zweryfikowano przez porównanie z wynikami otrzymanymi dla realizacji sprzętowych.

Rozdział VI zawiera propozycję uproszczonej metody analizy realizacji kodeków, która wykorzystuje modelowanie kolejkowe. Rozdział ten opisuje także przykładowe zastosowanie takiego modelowania.

W rozdziale VII przedstawiono analizę struktury optymalnej sieci w układzie, wykonaną dla podstawowych struktur NoC przeznaczonych dla dekoderów AVC. Szczegółowo omówiono w rozdziale wybór optymalnego rozwiązania.

Rozdział VIII przedstawia wnioski rozprawy i prezentuje wyników przeprowadzonych eksperymentów.

Rozdział 2

Zasadnicze wyniki rozprawy

2.1 Sieć w układzie – analiza korzyści i ograniczeń

Aby wskazać korzyści i ograniczenia sieci w układzie, przeprowadzono porównanie jej z innymi technikami wykonywania połączeń. Przy porównaniu wzięto pod uwagę dwie perspektywy: realizacji i projektowania. Porównanie właściwości przy implementacji sprzętowej podano w tabeli 2.1. Tabela ta zestawia ze sobą techniki połączeń bezpośrednich, współdzielonej szyny danych (magistrali) i sieci w układzie (NoC), w oparciu o doświadczenia autorki zdobyte w trakcie realizacji kodeka AVC. Natomiast tabela 2.3 porównuje właściwości projektowe dla połączeńbezpośrednich, magistrali i sieci.

Pierwszy wiersz w tabeli 2.1 porównuje wielkość interfejsu wymaganą przez wszystkie trzy techniki połączeń. Wartości podane w tabeli 2.2 przedstawiają wyniki syntezy dla modułów stosowanych w poszczególnych technikach. Rezultaty odnoszą się do zużycia zasobów sprzętowych w układze Xilinx Virtex-4 SX35. Proponowana sieć używa 8-bitowej szyny danych, a magistrale oparte są o łącza 32-bitowe. Magistrala z 8-bitową szyną danych po procesie syntezy wykorzystuje podobną ilość sprzętu (120 LUT i 11 FF) jak jednokierunkowy interfejsu NoC. Wykorzystanie tablic LUT w przypadku interfejsów NoC i routerów jest spowodowane przez realizację 32-bitowych kolejek wejściowych i wyjściowych jako rejestrów przesuwnych. Takie podejście ułatwia rozmieszczanie i trasowanie w końcowej wersji układu, ponieważ nie wymaga żadnego bloku pamięci, które mają ustalone położenie w układzie FPGA. Rozróżnienie na dwukierunkowy (BD) i jednokierunkowy (UD) interfejs sieci, jest podyktowane przez możliwość ograniczenia zużycia sprzętu. Nie wszystkie moduły przetwarzające

używają dwukierunkowego przesyłu danych, część z nich tylko wysyła, a część tylko odbiera dane. W związku z tym, w takich przypadkach nie ma potrzeby dołączania interfejsu dwukierunkowego. Sieć wprowadza koszt, który musi być uzasadniony przez oferowane przez nią możliwości w zakresie projektowania lub implementacji.

Pierwsza zaletą rozwiązania opartego na sieci jest łatwość rozmieszczania modułów i trasowania połączeń w fizycznym układzie przez narzędzie syntezy, ponieważ każdy moduł jest traktowany jako oddzielna część. Z tego powodu projektowanie jest bardziej elastyczne niż w przypadku połączeń bezpośrednich lub współdzielonej szynie danych. Aby uzyskać podobne efekty dla magistrali, wymagana jest segmentacja, co osiągnięto w prezentowanym kodeku poprzez zapewnienie dostępu do medium dla każdego modułu z interfejsem, który jest jednocześnie łącznikiem segmentu magistrali. Podobnie, w przypadku częstotliwości pracy, sieć jest skalowalna, a dodawanie nowych modułów przetwarzających do istniejącej infrastruktury nie ma wpływu na osiąganą częstotliwość pracy wynikowego układu. Magistrale współdzielone wykazują podobne właściwości, jeśli są gęsto segmentowane, jak w proponowanym kodeku. Ponieważ połączenia bezpośrednie są trudne do trasowania w układzie końcowym, narzędzie syntezy proponuje długie ścieżki połączeń, które charakteryzują się niską częstotliwością pracy.

Korzyści płynące z zastosowania sieci w układzie są również widoczne przy porównaniu możliwości projektowych pomiędzy trzema technikami połączeń, jak pokazano w tabeli 2.3. Chociaż przy wykorzystaniu sieci kosztem jest dłuższy czas projektowania niż w przypadku pozostałych technik, to osiąga się znaczne uproszczenie układu i przyspiesza proces wykrywania i usuwania błędów funkcjonalnych. Po pierwsze, rozwiązanie oparte na sieci umożliwia oddzielne debugowanie poszczególnych modułów, w oparciu tylko o jego dane wejściowe. Stan wyjść każdego modułu jest zależny jedynie od danych, które są odbierane przez sieć. Stąd każdy moduł może być sprawdzany na podstawie otrzymanych danych wejściowych.

Po drugie, użycie sieci pozwala na wdrożenie narzędzi, które umożliwiają przechwytywanie danych z pracującego układu do analizy poprawności danych przesyłanych przez sieć. Proponowane rozwiązanie oferuje możliwość transmisji do wielu odbiorców (multicast). Taki sposób transmisji, która jest używana do wysyłania jednego pakietu do wielu węzłów w sieci, pozwala przechwycić kopię pakietów przesyłanych przez sieć. Oznacza to, że możliwe jest przeprowadzenie analizy ruchu w układzie za pomocą zewnętrznego urządzenia jak w [14]. W proponowanym rozwiązaniu możliwe jest włączenie transmisji multicast dla każdego mo-

Tablica 2.1: Porównanie technik połączeniowych – implementacja sprzętowa

Właściwość	Połączenia bezpośrednie	Magistrala	Sieć
Rozmiar interfejsu	**Mały** – dostosowany do danego schematu komunikacji i formatu danych, jest włączany do PE.	**Większy** – standardowy interfejs jest modułem oddzielonym od PE, przetwarzającym dane przesyłane w ramach wiadomości na magistrali	**Największy** – standardowy interfejs jest modułem oddzielonym od PE, przetwarzającym dane przesyłane w pakiecie sieciowym. Jego rozmiar zależy od funkcji, które ma realizować
Rozmieszczanie i trasowanie w układzie scalonym	**Utrudnione** ze względu na dużą liczbę połączeń pomiędzy modułami, niektóre połączenia mogą być długie.	**Lepsze**, jednak wewnątrz jednego segmentu utrudnione jest trasowanie ze względu na rywalizację o pasmo, do którego ma dostęp kilka urządzeń.	**Dobre**, dzięki segmentacji. Narzędzie rozmieszczania i routingu traktuje każdy moduł osobno, i jest w stanie elastycznie przydzielać zasoby w układzie
Częstotliwość pracy	**Zmniejszona** z uwagi na rozmieszczanie i routing, które mogą generować długie połączenia między modułami, co znacznie zmniejsza częstotliwość pracy wynikowego układu	**Zależna od wielkości segmentu i wyniku rozmieszczania i trasowania.** Duże segmenty magistrali zmniejszają częstotliwość pracy wynikowego układu.	**Nie zmienia się** znacząco przy zmianie wielkości sieci.

Tablica 2.2: Wyniki syntezy dla elementów sieci. BD (ang. bi-directional) oznacza interfejs dwukierunkowy, UD (ang. uni-directional) oznacza interfejs jednokierunkowy

Technika połączeniowa	Moduł sieci	Liczba LUT	Liczba FF
Sieć w układzie (NoC)	router	430	330
	interfejs BD	310	190
	interfejs UD	110	100
szyna do/z mikroprocesora	nadajnik/odbiornik	217	11
szyna do/z pamięci	nadajnik/odbiornik	163	85

dułu w sieci. W rezultacie każdy pakiet, który jest przesyłany przez sieć z danego modułu jest kopiowany do routera i dalej do węzła przechwytującego, który przekazuje dane do urządzenia analizującego zlokalizowanego poza układem.

Po trzecie, wykorzystanie NoC pozwoliło na rozdzielenie w procesie projektowania i debugowania, układu parsującego/formującego strumień bitów AVC oraz predyktora, co wymagało zdefiniowania formatu danych przesyłanych między tymi urządzeniami. Ponadto obie części mogą być syntetyzowane niezależnie od siebie, co z kolei skraca jej czas. Synteza pojedynczej części trwała około 45 minut, podczas gdy synteza całości projektu trwała około 1,5 godziny na komputerze osobistym. Synteza taka była wykonywana wielokrotnie podczas debugowania, w celu sprawdzenia poprawności wprowadzanych zmian. Ponadto, usuwanie błędów może być przeprowadzane dla każdego modułu odzielnie. Wprowadzenie NoC skutkowało krótszym czasem projektowania i syntezy niż dla konkurencyjnych architektur połączeń.

Skalowalność systemu i mała liczba połączeń są kolejnymi korzyściami z systemów w układzie (Systems-on-Chip) opartymi o sieci. Pierwszym krokiem w budowaniu kodeka było uruchomienie dekodera, a następnie wykorzystanie go jako aplikacji bazowej rozszerzanej następnie o kodeka. Przejście od pierwszego do drugiego etapu wymaga dodawania nowych

urządzeń (np. estymacja ruchu). Dodanie urządzenia nie powodowało dużych zmian w projekcie, ponieważ protokół sieciowy został już zdefiniowany. Ponadto nie stwierdzono znacznego zmniejszenia przepustowości sieci, ponieważ nie występował problem rywalizacji o dostęp do medium transmisyjnego. Z drugiej strony nowe urządzenia generują ruch, który nie przekracza oferowanej przez sieć. Inną zaletą NoC jest mniejsza, niż w przypadku bezpośrednich połączeń, infrastruktura komunikacyjna (w szczególności liczba połączeń). Dzięki temu uzyskano łatwość debugowania i trasowania infrastruktury w wynikowym układzie scalonym.

Jedynym znaczącym kosztem projektu NoC okazał się czas na projektowania protokołu komunikacyjnego, formatów danych oraz urządzeń sieciowych. Jest to znacznie więcej niż w czasie projektowania połączeń bezpośrednich i nieco więcej niż w przypadku magistrali współdzielonej, jednak w rezultacie otrzymano większe możliwości diagnostyki błędów. Niemniej jednak czas ten odzyskiwany jest w procesie debugowania, który skracany jest przez dodatkowe funkcje diagnostyczne sieci. Uzyskuje się również krótki czas dodawania nowych modułów do SoC, ponieważ każdy nowy moduł musi mieć wcześniej dobrze zdefiniowaną funkcjonalność i formaty danych wejściowych i wyjściowych. Dodatkowo sieć NoC projektuje się raz, a następnie może być ponownie użyta w innym systemie SoC. Z tego punktu widzenia bezpośrednie połączenia słabo nadają się do wielokrotnego użytku.

Proponowana architektura współdzielonej szyny danych została zaprojektowana tak by oferować podobne możliwości jak sieć, co pociągnęło za sobą gęstą segmentację. Oba rozwiązania różnią się rodzajem transmisji. Na magistrali każdy segment jest zarezerwowany dla transmisji, a w sieci dane są przesyłane w postaci pakietów.

Tabela 2.4 przedstawia wyniki syntezy dla dekodera i kodera. Wskazany rozmiar układu parsującego lub formatującyego i predyktora nie obejmują rozmiaru sieci NoC. Dodanie zaproponowanej sieci oznacza zwiększenie rozmiaru całej konstrukcji o mniej niż 13%.

Rysunek 2.1 prezentuje działający układ kodeka. Proces kodowania i dekodowania jest wykonywany na płycie ML-402 [4]. Aby zapewnić obsługę strumienia wejściowego i wyjściowego, wykorzystano dwie karty Daughter Video [4]. Połączenie pomiędzy płytami zrealizowano z wykorzystaniem kabla programatora.

Tablica 2.3: Porównanie technik połączeniowych – projektowanie

Właściwość	Połączenia bezpośrednie	Magistrala	Sieć w układzie (NoC)
Liczba połączeń	**Wysoka** – każda komunikująca się para wymaga oddzielnych połączeń	**Zredukowana** – komunikacja przechodzi przez jedno wspólne połączenie	**Zredukowana** – komunikacja przechodzi przez jedno wspólne połączenie
Protokół	**Dostosowany do formatu danych** przesyłanych za pośrednictwem łącza	**Uniwersalny** – Rezerwacja segmentu na potrzeby transmisji	**Uniwersalny** – Transmisja pakietowa.
Prędkość łącza danych	**Wysoka** – połączenie nie jest współdzielone, a jego szerokość jest dostosowana do wymaganej szerokości pasma	**Zredukowana** – konkurencja o pasmo między urządzeniami w jednym segmencie, dane muszą być zawarte w wiadomości (oznacza to dodatkowy koszt - nagłówki i stopki), przetwarzanie wiadomości zmniejsza szybkość połączenia	**Zredukowana** – konkurencja o pasmo między urządzeniami w jednym segmencie, dane muszą być zawarte w pakiecie (oznacza to dodatkowy koszt - nagłówki i stopki), przetwarzanie pakietu zmniejsza szybkość połączenia.
Czas projektowania połączenia	**Krótki** – jednak każde nowe połączenie wymaga zaprojektowania protokołu komunikacji i formatu danych	**Dłuższy** – wymaga zaprojektowania protokołu i formatu danych, jednak projektowany raz, po dodaniu nowego modułu projektuje się tylko moduł przetwarzający dane na magistrali	**Dłuższy** – wymaga zaprojektowania protokołu i formatu danych, jednak projektowany raz, po dodaniu nowego modułu projektuje się tylko moduł przetwarzający dane na magistrali
Skalowalność projektu	**Mała** – dodawanie, usuwanie lub zastąpienie modułów wymaga przebudowy połączenia	**Średnia** – Dodawanie lub usuwanie modułu wymaga tylko dodania lub usunięcia interfejsu, jednak wpływa to na przepustowość (w jednym segmencie ze względu na konkurencję w dostępie do pasma)	**Wysoka** – zmiany w projekcie (dodawanie, usuwanie lub wymiana modułu) nie wpływa na pasmo ze względu na minimalną konkurencję w dostępie do pasma.
Możliwości wykrywania i usuwania błędów	**Małe**	**Średnie** – debugowanie zorientowane na połączenie, obniżona zdolność do przechwytywania wiadomości po wdrożeniu	**Wysokie** – debugowanie zorientowanych na połączenia, przechwytywanie pakiety po wdrożeniu

Tablica 2.4: Wyniki syntezy kodeka na układ Virtex4 SX35. Całkowity rozmiar układu jest sumą rozmiaru element PE i rozmiaru sieci.

Część kodeka		Rozmiar elementu przetwarzającego		Rozmiar sieci w układzie		Udział sieci	
Dekoder	Parser	10000 LUTs	6000 FFs	870 LUTs	730 FFs	9% of LUTs	12% of FFs
	Predyktor	10000 LUTs	9000 FFs	1090 LUTs	930 FFs	11% of LUTs	10% of FFs
Koder	Formater	10000 LUTs	6000 FFs	870 LUTs	730 FFs	9% of LUTs	12% of FFs
	Predyktor	16000 LUTs	14000 FFs	1510 LUTs	1220 FFs	9% of LUTs	9% of FFs

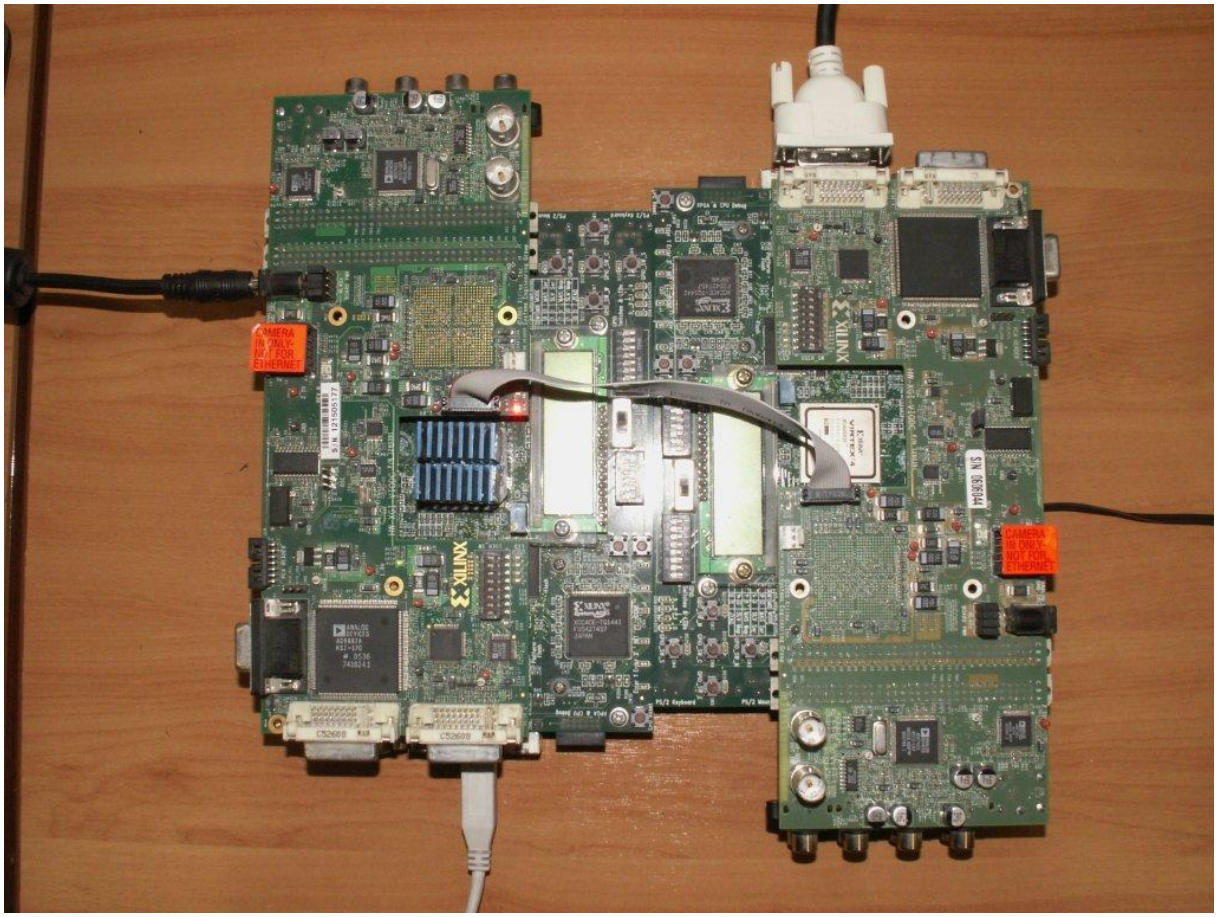
2.2 Zaproponowany symulator sieci w układzie

Modelowanie aplikacji sprzętowej można rozważyć na dwóch poziomach abstrakcji: aplikacji oraz platformy. Pierwsza opisuje zadania, które modelują funkcjonalność aplikacji, druga definiuje model platformy, na której wykonywane są zadania. Takie podejście pozwala na rozdzielenie mechanizmu symulacji od modelu oraz umożliwia symulację dowolonego modelu realizacji sprzętowej z wykorzystaniem tego samego oprogramowania do symulacji. W sekcji 5.3.2 rozprawy, autorka proponuje rozszerzenie opisu modelu, przedstawionego w [10], aby poprawić dokładność wyników generowanych przez symulator w odniesieniu do aplikacji przetwarzających dane wizyjne, w szczególności kodeki.

Dokument [10] stanowi specyfikację modelowania aplikacji i opisu sprzętu oraz zawiera wytyczne do tworzenia porównywalnych testów dla NoC. Został on stworzony przez Open Core Protocol International Partnership (OCP-IP), który jest niezależnym, niedochodowym konsorcjum branży przemysłu półprzewodnikowego. [10] opiera się na przeglądzie literatury dotyczącej sieci w układzie oraz modelowaniu aplikacji.

2.2.1 Model aplikacji

Symulator jest zgodny z głównymi zaleceniami modelowania aplikacji i strukturą pliku XML według [10]. W [10] określono rozdział aplikacji, platformy i opisu sieci w układzie. Takie podejście pozwala na symulację systemu SoC o dowolnej funkcjonalności i architekturze infrastruktury komunikacyjnej. Główną różnicą między proponowanym symulatorem a opisanym



Rysunek 2.1: Działający kodek AVC na dwóch płytach ML-402 [18]. Każda część znajduje się na osobnej płycie. Karta VIODC (Video IO Daughter Card [4]) obsługuje wejście i wyjście danych wizyjnych.

w rozdziale 5.2.3 rozprawy i w [10], jest elastyczny opis ruchu, który pozwala na modelowanie ilości danych przesyłanych w sieci w odniesieniu do rodzaju makrobloku i umożliwia uporządkowanie docelowych modułów w SoC. Modyfikacje te opisano w rozdziale 5.3.2 dysertacji.

Opis aplikacji zawiera zadania (tasks), które komunikują się ze sobą. Model obliczeń opisany w [10] jest podobny do sieci Kahn Process Network [8], dla której można utworzyć graf aplikacji. W takim grafie wierzchołki stanowią zadania obliczeniowe, a krawędzie reprezentują kanały komunikacyjne. Zadania są odwzorowywane na zasoby, aby określić, gdzie są wykonywane. Zasoby i model sieci są częścią platformy, która jest abstrkacyjne opisana za pomocą charakterystycznych parametrów sprzętowych.

Taki model zawiera opis struktury połączeń i rozmieszczenia elementów przetwarzających

w układzie. Struktura połączeń zawiera elementy modelujące urządzenia sieciowe takie jak routery i interfejsy. Urządzenia sieciowe i przetwarzające połączone są ze sobą za pomocą przewodów, podobnie jak w sieci sprzętowej. Elementy sieciowe dostarczają pakiety danych do modelowanych elementów przetwarzających (mPE). Routery zawierają tabelice routingu, które opisują, do którego portu wyjściowego należy skierować pakiet. Model interfejsu sieciowego (NI) odbiera pakiety przeznaczone do dołączonego mPE i rozpakowuje dane zawarte w nagłówku sieci i przekazuje je do właściwego zadania. Obliczenia wykonywane wewnątrz mPE modelowane są za pomocą zadań. Zadania odbierają dane i w zależności od ich ilości i rodzaju decydują jakie działania należy podjąć. Każdy symulowany element przetwarzający jest opisany tylko poprzez interakcje z siecią, czyli odbieranie i wysyłanie danych. To oznacza, że żadne z obliczeń wewnątrz mPE (tj. zadanie) nie są wykonywane, są modelowane jedynie jako stan uśpienia, w którym wykonują interakcje z siecią. Element jest traktowany jako czarna skrzynka charakteryzowana przez następujące rozkłady statystyczne opisujące oddziaływanie z siecią komunikacyjną:

- ilość danych potrzebnych do rozpoczęcia obliczeń,
- czas przetwarzania danych,
- ilość wysłanych danych.

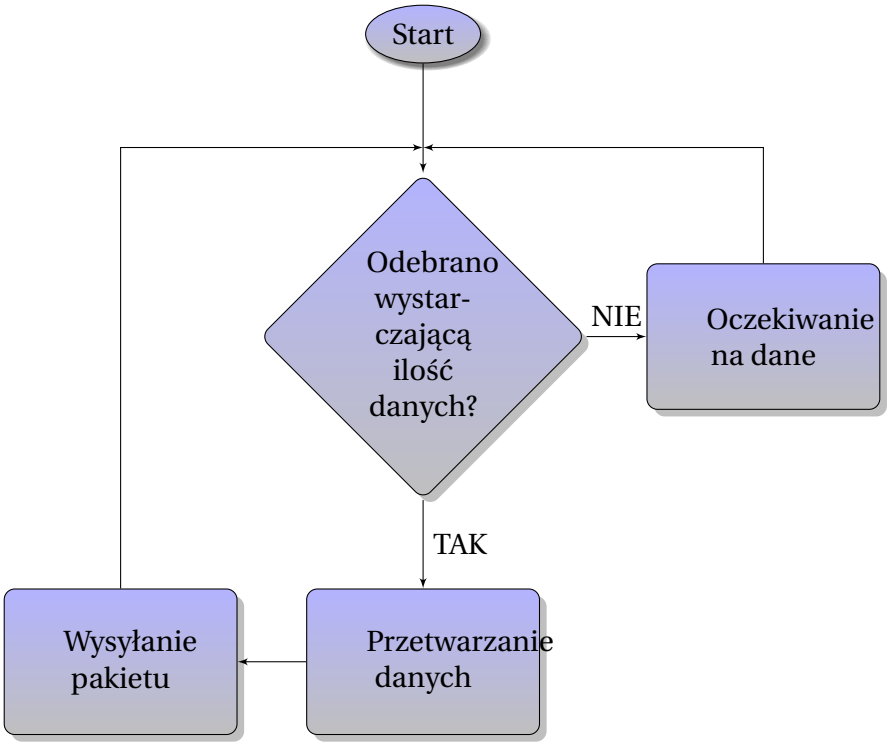
Rozkłady te są wykorzystywane przez każdy z mPE, który działa według schematu przedstawionego na rysunku 2.2.

Początkowo mPE sprawdza, czy istnieje wystarczająco dużo danych, aby rozpocząć obliczenia. Jeśli nie, mPE czeka na więcej danych. Jeśli jest wystarczająca ilość danych, mPE rozpoczyna przetwarzanie danych. W tym czasie mPE nie może odbierać danych z sieci. Po zakończeniu obliczeń, mPE wysyła pakiet i czeka na kolejne dane do przetwarzania. mPE odbiera dane z sieci za pomocą modelowanego interfejsu sieciowego (mNI), który buforuje odbierane i/lub wysyłane dane a logika sterująca zarządza interakcjami z siecią.

2.2.2 Proponowane zmiany modelowania aplikacji

Przykłady symulatorów przedstawionych w rozdziale 5.2.3 dysertacji oferują niewystarczające modele ruchu, który w przypadku dekodera sekwencji wizyjnej jest złożony. Według [10], każdy moduł jest modelowany na podstawie:

- ilości odebranych danych,



Rysunek 2.2: Schemat przejść między stanami mPE podczas symulacji

- ilości wysłanych danych,
- czasu przetwarzania.

Czasem przetwarzania w obliczeniach jest czas, w którym moduł nie odbiera ani nie wysyła żadnych danych. Każdy z wyżej wymienionych parametrów może być opisany jako rozkład statystyczny [10]. Niektóre propozycje rozwiązań, takie jak symulator opisany w [9], oferują możliwość zamiany rozkładu statystycznego na wartości rzeczywistych wysłanych danych. Metody te są jednak niewystarczające w przypadku modelowania kodeków. Pierwsze podejście proponuje zbyt prosty model, który nie zapewnia wystarczającego stopnia dokładności, a drugie wymaga dostarczania danych dla całej sekwencji wizyjnych, które mogą być również zasymulowane przy wykorzystaniu symulacji sprzętu. Co więcej, takie podejście również nie pozwala na uogólnienie modelu komunikacji, ale ogranicza się do określonej treści wizyjnej.

W trakcie przygotowywania modelu dekodera wizyjnego, autorka zauważyła, że modelowanie kodeka wizyjnego nie zależy od ilości danych (jak to zaproponowano w [10]), lecz od typu danych przesyłanych między modułami. Kolejno poczyniona została obserwacja, że

parametry niektórych modułów uzależnione są od prawdopodobieństwa warunkowego, w szczególności czas obliczeń zależy od poprzednio wybranej wartości w mikroprocesorze. Te dwie obserwacje omówiono w następnych akapitach.

Rodzaj danych przesyłanych pomiędzy modułami

Dekoder przetwarza dane zależnie od typu makrobloku, który definiuje zakres wartości dla czasu obliczeń i ilość danych wejściowych i wyjściowych, a także określa moduły, które uczestniczą w dekodowaniu. W szczególności moduł do predykcji intra nie uczestniczy w dekodowaniu makrobloków P i B. Ponadto moduł predykcji inter jest wyłączony z dekodowania makrobloków I. Takie zachowanie elementów jest trudne do modelownia w oparciu wyłącznie o ilość danych przesyłanych między modułami, zwłaszcza jeśli symulacja musi zamodelować system z różnymi typami makrobloków w jednej ramce. Ponadto istnieją różne sposoby podziału każdego typu makrobloku, narzucające ilość danych i czas przetwarzania makrobloku w każdym module. Ponadto, w przypadku makrobloków P i B, rodzaj segmentacji określa ilość danych konteksu, które muszą być pobrane z pamięci. Te czynniki należy brać pod uwagę przy tworzeniu modelu kodeka, niezależnie od jego realizacji sprzętowej.

Warunkowe prawdopodobieństwo czasu obliczeń

Przetwarzanie sekwencji zawierajacej ten sam typ danych może prowadzić do zmiany czasu przetwarzania dla kolejnych porcji danych. Może to nastąpić zwłaszcza w mikroprocesorze, gdzie rozpoczęcie takiej sekwencji danych jest poprzedzone przeładowaniem buforów dla programu i danych, podczas gdy dla kolejnych bloków danych wymagane jest jedynie przeładowanie pamięci danych. Ponadto przeładowanie buforów danych pokrywa się z przetwarzaniem danych, powodując dalsze skrócenie czasu obliczeń. W takiej sytuacji pojedynczy rozkład statystyczny nie modeluje czasu przetwarzania z wystarczającą dokładnością. Dodatkowo, należy wykorzystać informacje o wcześniej przetwarzanych danych. Biorąc pod uwagę przedstawione okoliczności, autorka zdecydowała się na wprowadzenie możliwości modelowania ilości danych oraz czasu obliczeń z wykorzystaniem prawdopodobieństwa warunkowego. Ich rozkład zależy od liczby wcześniej przetworzonych bloków danych.

Proponowany symulator NoC pozwala określić rozkłady statystyczne, które odwzorowują cechy dekodera wizyjnego. Pozwala opisać reakcję modułu w zależności od różnych typów

danych, które może otrzymać. Ponadto czas obliczeń modułów i ilość danych wyjściowych można modelować za pomocą prawdopodobieństwa warunkowego. Dodatkowo każdy moduł może naśladować tę samą sekwencję procedur jak w rzeczywistej sprzętowej realizacji tj. śledzenie. Oznacza to, że jeśli moduł, w przypadku makrobloku typu X, przesyła dane najpierw do modułu A, następnie do B i kolejno do C, to ta sama sekwencja jest rekonstruowana w symulatorze. Ponadto jeśli ta sekwencja zależy od typu makrobloku, to symulator jest również w stanie odtworzyć taką zmianę sekwencji. Każdy z wymienionych przypadków charakteryzuje się własnym rozkładu statystycznym. Wybór aktualnie dekodowanego typu makrobloku jest oparty na losowaniu z rozkładu opisującego prawdopodobieństwo wystąpienia dla każdego typu.

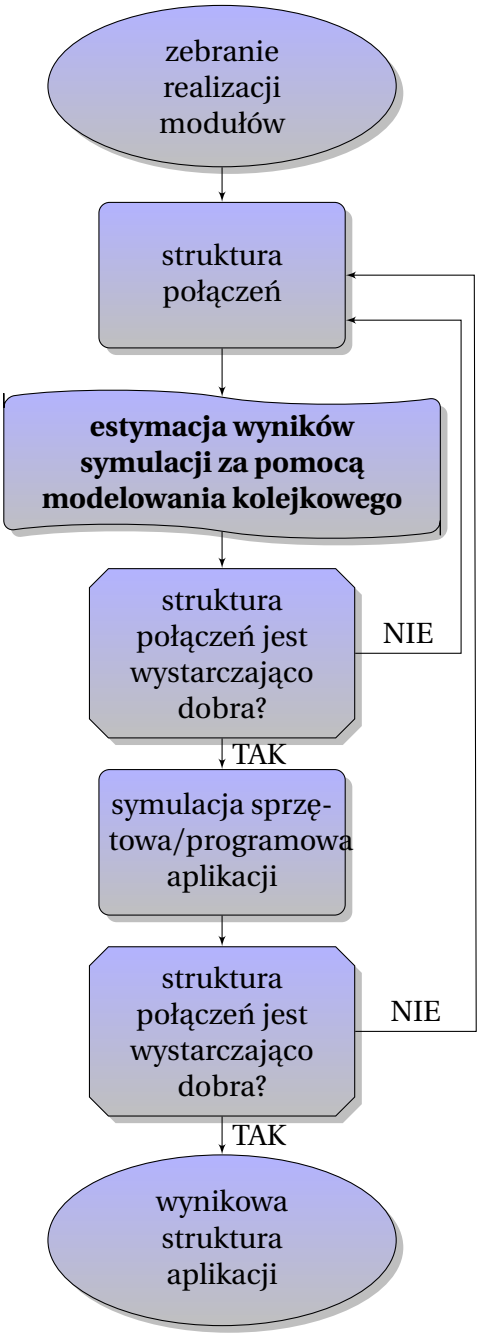
2.3 Modelowanie kolejkowe w projektowaniu sieci w układzie

2.3.1 Proponowany przebieg projektowania

Typowy schemat przebiegu projektowania realizacji sprzętowej przedstawiono na rysunku 1.1 rozprawy. Składa się on z wybrania rdzeni IP i ich statystyk, wybrania architektury połączeniowej i sprawdzania jej za pomocą symulacji programowej bądź sprzętowej. Jeśli wybrana architektura nie spełnia wymogów, wybierana jest inna architektura i sprawdzenie wykonywane jest powtórnie. Proces ten trwa aż do uzyskania zadowalającego rozwiązania.

Przedstawiony alogrytm projektowania jest czasochłonny, choć użycie symulacji programowej skraca czas potrzebny na ocenę struktury połączeń. Również wyniki samej symulacji podlegają dalszemu przetwarzaniu i analizie w celu oszacowania możliwości danej architektury. Przybliżenie wyników symulacji może być uzyskane przy użyciu modelowania kolejkowego, z dokładnością umożliwiającą porównanie między sobą różnych architektur połączeń komunikacyjnych. Nowy przepływ procesu projektowania, z uwzglednieniem nowego kroku zaproponowano na rysunku 2.3. Pierwsze dwa kroki w zmodyfikowanym procesie pozostają takie same (wybór opisów statystycznych modułów i wybór architektury). Te dwa kroki wskazują na konieczność zbudowania dwóch grafów: opisu aplikacji i topologii, jak przedstawiono w rozdziale I rozprawy. Mając na uwadze takie właściwości sieci, można wybrać odpowiedni model kolejkowy. Jak opisano w rozdziale 6.1 dysertacji, najprostszym modelem kolejki jest M/M/1, przyjęty tu do dalszych obliczeń. Główną zaletą proponowanego przebiegu projek-

towania (patrz rysunek 2.3) jest to, że wewnętrzna pętla jest szybka i dzięki temu może być wykonywana wielokrotnie. Natomiast czasochłonne symulacje wykonywane są rzadko.



Rysunek 2.3: Schemat procesu projektowania z zaznaczonym nowym etapem

2.3.2 Modelowanie systemu kolejkowego i obliczenia

Według [11] w celu przeprowadzenia symulacji aplikacji sprzętowej, należy dostarczyć jej opis w postaci: grafów aplikacji (tzn. przepływ danych pomiędzy modułami i czas obliczeń dla każdego modułu) i graf topologii (który opisuje, jak rozmieszczone są moduły i jaka jest charakterystyka routerów). Te same dane potrzebne są do modelowania kolejki. Celem modelowania kolejkowego jest oszacowanie opóźnienia, które jest wymagane do zdekodowania makrobloku.

Opóźnienie jest liczone wzdłuż ścieżki dekodowania, oddzielnie dla każdego typu makrobloku w ramce. Wyniki przybliżają średni czas przetwarzania makrobloku i takie średnie wartości są wykorzystywane do następnych obliczeń. Wyniki są wyrażone w cyklach zegarowych.

2.3.3 Korzyści i ograniczenia proponowanego systemu modelowania kolejkowego

Wykonana analiza opóźnień wskazuje kilka zalet modelowania kolejkowego systemu SoC. Pierwszą jest logiczny podział na ścieżki przetwarzania danych i analizę opóźnień osobno dla poszczególnych ścieżek. Taki sposób analizy pozwala na identyfikację w systemie miejsc o ograniczonej przepustowości. Wyniki obliczeń wskazują również jak bardzo strumień zgłoszeń odbiega od strumienia obsługi. W przypadku np. modułu predykcji wartości są zadowalające, jednak mikroprocesor wprowadza opóźnienie, które warunkuje czas przetwarzania makrobloku. Oznacza to również, że parser strumienia bitowego wymaga dalszej optymalizacji. Skala opóźnień parsera jest w dużej mierze zależna od standardu kompresji AVC[1], który ogranicza możliwość zrównoleglenia dekodowania strumienia bitowego.

Proponowane zastosowanie modelowania kolejkowego dla SoC pozwala na rozdzielenie opóźnienia wprowadzonego przez moduły na ścieżce danych od czasu potrzebnego na przykład na pobranie danych z pamięci. Pozwala to na oszacowanie na ile dostęp do pamięci opóźnia proces dekodowania.

Chociaż modelowanie kolejkowe pozwala na elastyczną analizę wydajności IP Core, to nadal nie może zastąpić symulacji w szczegółowej analizie danej realizacji sprzętowej. Jednakże można wskazać przydatność modelowania kolejkowego nawet przy analizie z użyciem najprostszego modelu.

Rozdział 3

Podsumowanie

3.1 Oryginalne osiągnięcia rozprawy

Oryginalnym osiągnięciem pracy jest propozycja architektury NoC (w zespole badawczo-rozwojowym), która jest osiągnięciem projektowym oferującym wysoką wydajność przy ponoszonych niskich kosztach sprzętu.

Zaproponowane rozwiązanie opiera się na topologii pierścienia i wykorzystaniu interfejsów sieciowych z możliwością przełączania w celu zmniejszenia zużycia zasobów sieci. Routery są stosowane do oddzielenia ruchu i zapewaniają buforowanie pomiędzy częściami aplikacji sprzętowej. Proponowana architektura jest korzystna zwłaszcza dla małych aplikacji, takich jak kodeki, ponieważ pozwala wykorzytsać możliwości NoC przy koszcie porównywalnym do kosztu współdzielonej szyny danych.

Autorka udowodniła stosowalność NoC dla kodeków w oparciu o przykładową realizację kodeka AVC, mimo niewielkiego rozmiaru takiego układu. Korzyści, które uzyskano dotyczą uproszczenia projektowania i weryfikacji. Skalowalność architektury NoC pozwala na dalszy rozwój aplikacji dekodera i wzbogacenie go o możliwości kodowania. Ponadto proces projektowania zorientowany na połączenia ułatwił proces debugowania i oddzielił projektowanie funkcji układu od infrastruktury komunikacyjnej, co pozwoliło na weryfikację modułów w oparciu o poprawność sygnałów wejściowych i wyjściowych.

Ważnym osiągnięciem tej pracy jest propozycja modelowania struktury zaawansowanych kodeków dla symulatorów NoC. Uproszczony model pozwala na symulację i analizę wyników wydajności dla dowolnej topologii i szerokiego zakresu aplikacji przetwarzania sekwencji wi-

zyjnych. Autorka przedstawiła przykładowe wyniki modelowania dla dekodera AVC. Symulator został wykorzystany również do zbadania różnych propozycji topologii. Chociaż doświadczenia dostarczyły podobne wyniki dla różnych struktur sieciowych, to wskazują one na korzyści wynikające z rozdzielenia ruchu, nawet dla małych konstrukcji, takich jak dekoder AVC.

Innym oryginalnym osiągnięciem jest propozycja estymacji wyników symulacji z wykorzystaniem prostych obliczeń. Do realizacji tego celu wykorzystano modelowanie kolejkowe a otrzymane wyniki pozwoliły na zgrubną ocenę wydajności systemu. W rezultacie zaproponowano przyjęcie modelowania kolejkowego jako narzędzia wstępnej analizy systemu, jednak nie jako zamiennik symulacji. Użycie tego narzędzia pozwala na oszacowanie opóźnień wprowadzanych przez poszczególne moduły, a także na ścieżkach opóźniających. Daje także możliwość dostosowania własności bloków funkcjonalnych na etapie ich wyboru i wykrycia ewentualnych miejsc w aplikacji o zmniejsoznej przepustowości.

3.2 Wnioski ogólne

Rozprawa wykazała przydatność architektury NoC dla kodeków sekwencji wizyjnych. Korzyści i ograniczenia można uogólnić na wiele zastosowań, szczególnie tych, dla których planuje się dalszą rozbudowę (np. z dekodera do enkodera). Analiza bazuje na realizacji sprzętowej kodera AVC, a wyniki doświadczeń mają znaczenie dla szerokiego zakresu zastosowań, obejmujących implementacje nowych standardów kodowania sekwencji wizyjnych, transkoderów i narzędzi do analizy sygnały wizyjnego. Każde z wymienionych zastosowań może być modelowane za pomocą zaproponowanych narzędzi modelowania i zasymulowane za pomocą zaproponowanego symulatora. Proponowany sposób modelowania ruchu może być stosowany do dowolnego rodzaju aplikacji i jest szczególnie odpowiedni dla modułów o złożonej charakterystyce wymiany danych.

Wyniki symulacji wykazały podobną wydajność systemu dla różnych architektur, co dowodzi, że NoC charakteryzuje się pojemnością przekraczającą wymagania danej aplikacji. Niemniej jednak uzyskane wyniki wskazują, że najlepsze efekty uzyskuje się dla architektury NoC, oferującej rozdzielenie ruchu pomiędzy poszczególnymi częściami aplikacji. Pokazują one także, że infrastruktura komunikacyjna dla układów sprzętowych składających się z około 10 modułów wymaga już rozdzielenia ruchu.

Autorka wykazała przydatność modelowania kolejkowego do zastosowań sprzętowych i

sposób jego wykorzystania do estymacji opóźnienia. Przedstawione obliczenia mogą być zastosowane do dowolnej realizacji sprzętowej ze strumienień zgłoszeń o rozkładzie Poissona i wykładniczym strumieniu obsługi. Jednakże uzyskane wyniki wskazują, że nawet w przypadku gdy aplikacja nie może zostać opisana w taki sposób, można zastosować do analizy inne odpowiedniejsze modele kolejek. Rozprawa wykazała przydatność takiego modelowania.

Bibliografia

[1] ISO/IEC 14496-10, Generic Coding of Audio-Visual Objects, Part 10: Advanced Video Coding.

[2] Audio Video Coding Standard Workgroup of China (AVS),. *The Standards of Peopleâ€™s Republic of China GB/T 20090.2-2006, Information Technology â€“ Advanced Coding of Audio and Video â€“ Part 2:Video*, 2006.

[3] VC-1 Compressed Video Bitstream Format and Decoding Process. *Society of Motion Picture and Television Engineers*, (SMPTE 421M-2006), 2006.

[4] XILINX Inc., UG239, Video Starter Kit Quick Start Guide. (0402447):1–8, 2006.

[5] J.-R. B. Bross, W.-J. Han, and T. W. Ohm, G. J. Sullivan. High Efficiency Video Coding (HEVC) text specification Working Draft 5. In *Joint Collaborative Team on Video Coding (JCT-VC) of ITUT SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-G1103*, Geneva, 2011.

[6] M. Domański. *Obraz cyfrowy*. Wydawnictwa Komunikacji i Łączności, 1 edition, 2010.

[7] J. Flich, S. Rodrigo, J. Duato, T. Sø dring, A. s. G. n. Solheim, T. Skeie, and O. Lysne. On the Potential of NoC Virtualization for Multicore Chips. In *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 801–807. IEEE, Mar. 2008.

[8] T. Kangas, J. Riihimaki, E. Salminen, K. Kuusilinna, and T. Hamalainen. Using a communication generator in SoC architecture exploration. In *Proceedings. 2003 International Symposium on System-on-Chip (IEEE Cat. No.03EX748)*, pages 105–108. IEEE, 2006.

[9] M. Lis, K. Shim, M. Cho, P. Ren, and O. Khan. DARSIM: a parallel cycle-level NoC simulator. In L. E. Wenisch and Thomas, editors, *6th Annual Workshop on Modeling, Benchmarking and Simulation*, Saint Malo, France, 2010.

[10] E. Salminen, G. Cristian, T. D. Hamalainen, and A. Ivanov. Network-on-Chip Benchmarking Specification Part 1 : Application modeling and hardware description. Technical report, OCP International Partnership, Beaverton, OR 97006 USA, 2008.

[11] E. Salminen, A. Kulmala, and T. Hamalainen. Survey of Network-on-chip Proposals. Technical Report March, OCP International Partnership, Beaverton, OR 97006 USA, 2008.

[12] V. Soteriou and N. Eisley. Polaris: A System-Level Roadmapping Toolchain for On-Chip Interconnection Networks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(8):855–868, Aug. 2007.

[13] M. Stepniewska, A. Luczak, and J. Siast. Network-on-Multi-Chip (NoMC) for Multi-FPGA Multimedia Systems. *2010 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools*, pages 475–481, Sept. 2010.

[14] M. Stepniewska, O. Stankiewicz, A. Luczak, and J. Siast. Embedded debugging for NoCs. In *Mixed Design of Integrated Circuits and Systems (MIXDES), 2010 Proceedings of the 17th International Conference*, pages 601–606, 2010.

[15] G. Sullivan, J.-R. Ohm, W.-j. Han, and T. Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE TRANS. ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY*, PRE-PUBLIC, 2012.

[16] J. Tyszer. *Object-Oriented Computer Simulation of Discrete-Event Systems*. Springer, 1999.

[17] P. Vandewalle, J. Kovacevic, and M. Vetterli. Reproducible research in signal processing. *IEEE Signal Processing Magazine*, 26(3):37–47, May 2009.

[18] Xilinx. *Xilinx UG080 ML401/ML402/ML403 Evaluation Platform User Guide*, volume 080. 2006.