AUTOMATYKA • 2008 • Tom 12 • Zeszyt 3

Agnieszka Dąbrowska-Boruch*, **, Kazimierz Wiatr*, **

Implementacja kodeka standardu MPEG-2 w układach FPGA

1. Wprowadzenie

Standard MPEG-2 jest obecnie jednym z najpowszechniej używanych standardów kompresji sekwencji obrazów wizyjnych. Najczęściej wykorzystywany jest on w telewizji cyfrowej. Standard ten jest także stosowany w kamerach cyfrowych, na płytach DVD, w tunerach cyfrowych itp.

Podstawowym wymaganiem postawionym przed standardem MPEG-2 jest dążenie do osiągnięcia równocześnie jak największego stopnia kompresji oraz możliwość osiągnięcia jak największej jakości dekodowanej sekwencji wizyjnej przy założonej szybkości transmisji. Dodatkowo wymagana jest kompatybilność ze standardem MPEG-1. MPEG-2 nie standaryzuje kodowania strumienia wizyjnego, standaryzowana jest składnia strumienia wizyjnego oraz semantyka dekodowania.

W standardzie MPEG-2 [2–9] rozróżniane są dwie techniki kodowania: kodowanie wewnątrzobrazowe oraz kodowanie międzyobrazowe. Metody kodowania wewnątrzobrazowego wykorzystują przestrzenną redundancję istniejącą pomiędzy sąsiednimi pikselami w danym obrazie. Tak zakodowane obrazy są to tak zwane obrazy typu I (*Intraframe* lub *Intrapicture*). Stosowanie obrazów tego typu zapewnia dostęp do dowolnej części skompresowanej sekwencji. Dodatkowo obrazy typu I można stosować przy zmianie sceny oraz w innych przypadkach, gdzie kompensacja obrazu jest nieefektywna. Algorytm kompresji bazuje na dwuwymiarowej dyskretnej transformacji kosinusowej. Współczynniki otrzymane z 2D-DCT poddawane są następnie procesowi kwantyzacji. Podobnie jak w standardzie JPEG, również w MPEG-2 określone są typowe macierze kwantyzacji dla współczynników transformacji kosinusowej. W wyniku kwantowania amplituda wielu współczynników jest równa zero. Kompresja jest osiągana poprzez kodowanie entropowe amplitudy niezerowych współczynników oraz zajmowanej przez nie pozycji w bloku.

Aby umożliwić swobodny dostęp do dowolnego punktu sekwencji, obrazy typu I powinny pojawiać się stosunkowo często. Jednak w związku z tym, że obrazy tego typu nie

* Katedra Elektroniki, Akademia Górniczo-Hutnicza w Krakowie
** ACK CYFRONET, Akademia Górniczo-Hutnicza w Krakowie

wykorzystują korelacji czasowej, ich stopień kompresji jest mały. Aby zwiększyć efektywność kompresji takiej sekwencji, należy wstawić pomiędzy dwa kolejne obrazy typu I kilka obrazów innego typu. Zbyt duża liczba takich dodatkowych obrazów powoduje utratę swobodnego dostępu do dowolnego punktu sekwencji.

W standardzie MPEG-2 występują jeszcze dwa typy obrazów:

1) obrazy typu P zakodowane predykcyjnie (*Predictive coded pictures*),
2) obrazy typu B zakodowane predykcyjnie dwukierunkowo (*Bidirectinally-predictive coded pictures*).

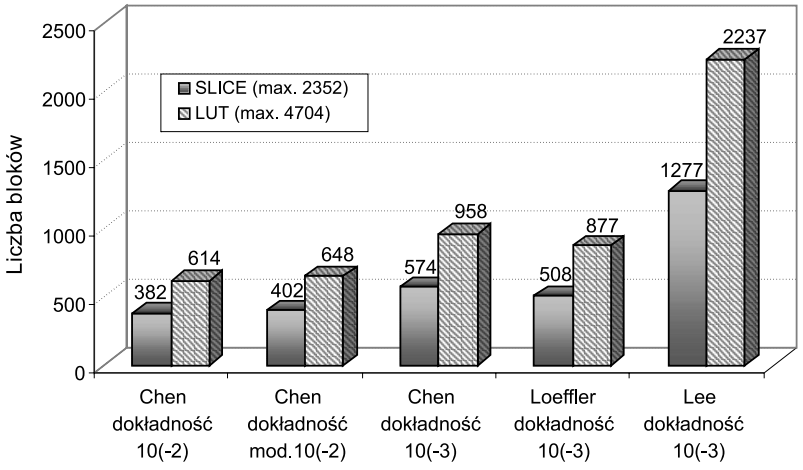
W sekwencji obrazów, oprócz przestrzennej redundancji występuje również redundancja czasowa wynikająca z wysokiego stopnia korelacji pomiędzy kolejnymi obrazami. Do obliczania błędu predykcji w algorytmie standardu MPEG-2 wykorzystywana jest redundancja czasowa.

2. Implementacje algorytmów transformacji DCT i IDCT

Podstawowy algorytm jednowymiarowej dyskretnej transformacji kosinusowej, który przetwarza wektor złożony z 8 pikseli, wykorzystuje 64 operacje mnożenia oraz 56 operacji dodawania. Z porównania zasobów sprzętowych układu programowalnego FPGA wykorzystywanych przez układ mnożący i przez układ dodający wynika, że mnożenie jest operacją wymagającą więcej zasobów układu programowalnego niż operacja dodawania. Z punktu widzenia sprzętowej implementacji korzystniejsze jest zatem zastosowanie algorytmu DCT o zredukowanej liczbie operacji mnożenia kosztem nieznacznego zwiększenia liczby operacji dodawania. Przy realizacji algorytmu dyskretnej transformacji kosinusowej, ważnym aspektem jest również dokładność otrzymywanych wyników, która w znaczący sposób wpływa na dokładność rekonstrukcji skompresowanego obrazu. W związku z tym, głównym kryterium decydującym o wyborze algorytmu dyskretnej transformacji kosinusowej, przy zbliżonych parametrach sprzętowej implementacji i przy tej samej dokładności odwzorowania współczynników mnożenia, była minimalizacja błędu pomiędzy wynikami analitycznymi a implementacją sprzętową. Z punktu widzenia działania systemu, w skład którego wchodzi blok dyskretnej transformacji kosinusowej, ważna jest również szybkość przetwarzania danych wejściowych. Biorąc pod uwagę powyższe zagadnienia, można stwierdzić, że najkorzystniejszym rozwiązaniem jest zastosowanie algorytmu z układami mnożącymi o różnej dokładności odwzorowania współczynników mnożenia, stanowiącego kompromis pomiędzy szybkością przetwarzania a dokładnością otrzymywanych wyników. O tym, które współczynniki zostały zaimplementowane z mniejszą dokładnością, decydował ich wpływ na wartości otrzymywane w wyniku dyskretnej transformacji kosinusowej. Opracowana przez autorów modyfikacja polega na implementacji współczynnika y_0 z dokładnością 10^{-3} i pozostałych współczynników z dokładnością 10^{-2}. Algorytm ten, opracowany przez autorów pracy, został użyty do skonstruowania dwuwymiarowej DCT.

Powyższa modyfikacja pozwoliła na znaczną redukcję zajętości zastosowanego układu programowalnego FPGA, przy nieznacznym pogorszeniu otrzymanych wartości wyjściowych, przy czym różnica pomiędzy pikselem zrekonstruowanym a jego pierwowzorem

pozostała niezauważalna. Ponadto należy zauważyć, że modyfikacja algorytmu Chena (rys. 1), zaproponowana przez autorów i uwzględniająca te założenia, pozwala na sprzętową implementację zapewniając jednocześnie małą różnicę pomiędzy obrazem pierwotnym a obrazem zdekompresowanym.



Rys. 1. Zajętość zasobów układu XCV200BG352 w zależności od implementowanego algorytmu i dokładności odwzorowania współczynników mnożenia

3. Implementacje operacji kwantyzacji i dekwantyzacji

Głównym założeniem uczynionym w trakcie implementacji procesów kwantyzacji oraz dekwantyzacji było zapewnienie zgodności funkcjonalnej określonej przez specyfikację standardu MPEG-2.

Z punktu widzenia wydajności przetwarzania danych wejściowych, ważnym aspektem było zrównoleglenie wykonywanych obliczeń. Zaimplementowane bloki kwantyzatora oraz dekwantyzatora przetwarzają wektory złożone z 8 danych wejściowych, a dane na wyjściu bloków pojawiają się co każdy cykl zegara. Takie zrównoleglenie obliczeń miało na celu przystosowanie bloków kwantyzacji/dekwantyzacji do przetwarzania danych dostarczanych z procesów 2D-DCT i 2D-IDCT. Dane te są wektorami złożonymi z 8 wartości. Taki sposób implementacji procesu kwantyzacji i dekwantyzacji zwiększa zajętość wykorzystywanych zasobów, lecz równocześnie zwiększa szybkość przetwarzania. Zostały zaproponowane dwa sposoby realizacji procesów kwantyzacji i dekwantyzacji. Pierwszy z nich wykorzystuje bloki o stałym współczynniku mnożenia. Bloki te zostały tak zaimplementowane, aby zajmowały jak najmniej zasobów. Drugim sposobem realizacji jest wykorzystanie bloków LUT oraz wbudowanych mnożarek o rozmiarach 18×18. Pod względem wydajności najlepsze jest rozwiązanie wykorzystujące mnożarki o rozmiarach 18×18 wbu-

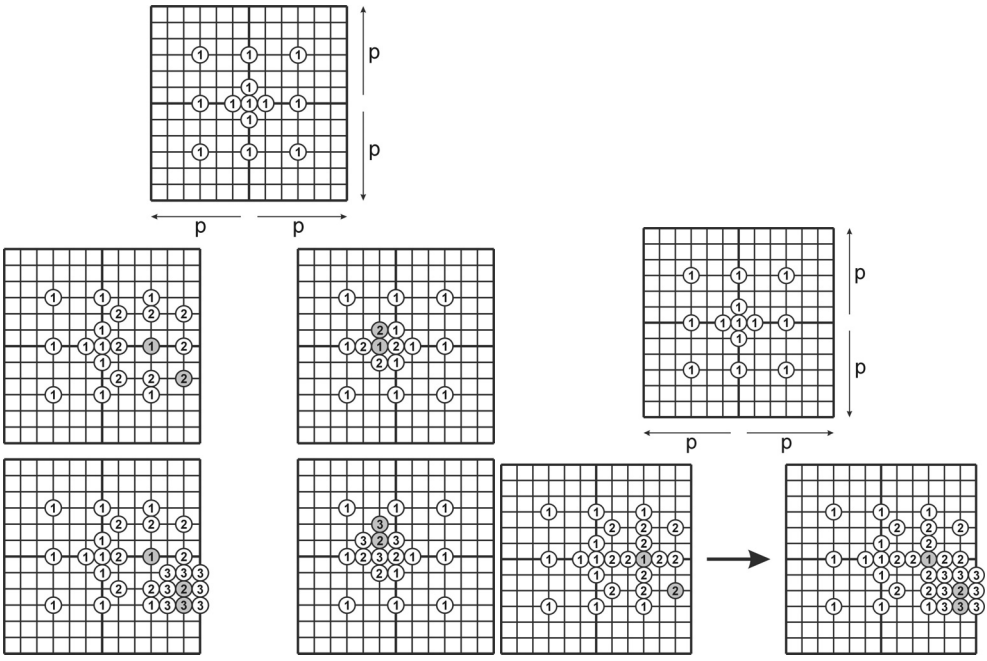
dowane w układzie Virtex-II PRO firmy Xilinx. Tak zaimplementowany blok kwantyzacji i dekwantyzacji jest w stanie przetworzyć znacznie więcej danych wejściowych niż w rozwiązaniu opartym na mnożarkach ze stałym współczynnikiem mnożenia. Jednak, jeśli docelowy układ, w którym zostaną zaimplementowane procesy kwantyzacji i dekwantyzacji, nie będzie miał możliwości wykorzystania wbudowanych mnożarek, układy mnożące będą realizowane w oparciu o oferowaną przez układ logikę programowalną. Taka realizacja bloków kwantyzacji i dekwantyzacji jest rozwiązaniem uniwersalnym dla wszystkich układów programowalnych FPGA. Pomimo że jest to rozwiązanie mogące przetworzyć mniejszą liczbę danych wejściowych, to przy implementacji w układzie XCV2P125FF1704 firmy Xilinx, blok procesu kwantyzacji może przetwarzać w ciągu sekundy 842 obrazy o rozdzielczości 1920×1152 pikseli, a blok procesu dekwantyzacji może przetwarzać w ciągu sekundy 837 obrazów o rozdzielczości 1920×1152 pikseli. Zatem bloki te mogą być wykorzystywane w systemie pracującym w czasie rzeczywistym.

4. Implementacje procesu estymacji ruchu

Estymacja ruchu wchodząca w skład toru kompresji sekwencji obrazów ruchomych, wykorzystującego standard MPEG-2, jest operacją najbardziej złożoną pod względem obliczeniowym, a zatem jest najbardziej czasochłonna. Najkorzystniejsze jest zatem jak największe zrównoleglenie tych obliczeń. Rozwiązaniem problemu ze zrównolegleniem obliczeń jest zastosowanie wewnętrznej pamięci BRAM układu programowalnego FPGA oraz zwielokrotnienie bloku odpowiedzialnego za obliczanie funkcji celu.

Z punktu widzenia dokładności otrzymywanych wyników najlepszym algorytmem jest algorytm pełnego przeszukiwania FS [11], jednak wymaga on wyznaczenia funkcji celu dla każdej lokacji w obszarze przeszukiwania. W związku z tym, zastosowane zostały algorytmy o mniejszej złożoności obliczeniowej. Algorytmem, który jest rekomendowany do wykorzystania przy sprzętowej implementacji kodera standardu MPEG-2, jest algorytm trójkrokowego przeszukiwania TSS [11]. Na bazie tego algorytmu powstał algorytm efektywnego trójkrokowego przeszukiwania E3SS [12]. Algorytm ten jest bardziej dokładny od swojego pierwowzoru. Pomimo większej zajętości zasobów układu programowalnego FPGA niż algorytm TSS, implementacja algorytmu efektywnego trójkrokowego przeszukiwania E3SS jest lepszym rozwiązaniem pod względem dokładności otrzymywanych wyników oraz zdolności przetwarzania danych wejściowych. Jednak wadą tego algorytmu jest duża liczba możliwych etapów. W związku z tym, zaproponowane zostały modyfikacje polegające na redukcji liczby etapów działania algorytmu E3SS oraz uproszczenia logiki sterującej przy jednoczesnym wykorzystaniu wszystkich bloków wyznaczających funkcje celu. Dzięki tym modyfikacjom udało się osiągnąć mniejszą zajętość zasobów układu programowalnego FPGA oraz możliwość przetwarzania większej liczby danych wejściowych, niż miało to miejsce w przypadku zaimplementowanych standardowych algorytmów TSS oraz E3SS.

W związku z tym, że zaimplementowane algorytmy estymacji ruchu nie narzucają, z jakich funkcji celu należy korzystać przy wyznaczaniu najlepszego dopasowania, powstaje problem, która funkcja celu jest najodpowiedniejsza przy sprzętowej realizacji wybranego algorytmu. W związku z powyższym, do celów sprzętowej realizacji została wybrana funkcja klasyfikacji różnic pomiędzy pikselami PDC. Funkcja ta jest kompromisem pomiędzy złożonością sprzętowej implementacji a dokładnością otrzymywanych wyników.

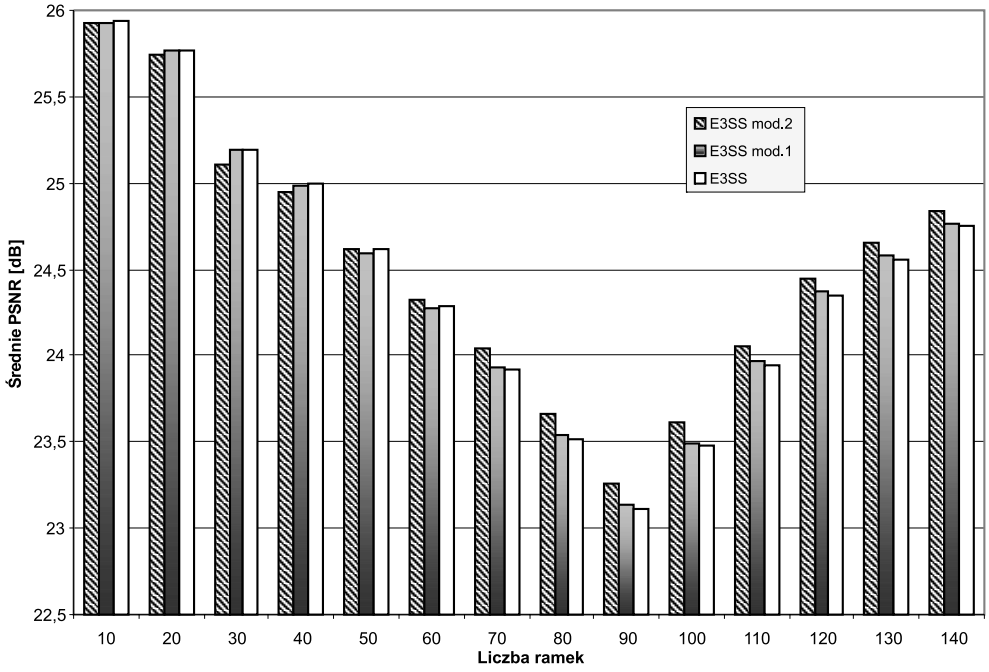


Rys. 2. Modyfikacja algorytmu E3SS z ograniczeniem do trzech etapów – „E3SS mod.1" (z lewej) oraz z ograniczeniem do trzech etapów i wykorzystaniem wszystkich jednostek obliczających funkcje celu – „E3SS mod.2" (z prawej)

Zaletą zaproponowanej modyfikacji „E3SS mod.2" jest taki sam maksymalny czas wymagany do przetworzenia danych wejściowych jak w przypadku poprzedniej modyfikacji (rys. 2). Dodatkowym atutem tego algorytmu jest prostsza sprzętowa implementacja logiki sterującej działaniem algorytmu oraz wykorzystanie wszystkich dostępnych jednostek odpowiadających za obliczanie funkcji celu dla poszczególnych lokacji. Jednak algorytm „E3SS mod.2" w porównaniu z E3SS uzyskuje korzystniejszy szczytowy stosunek sygnału do szumu w przypadku sekwencji cechujących się ruchem na małym obszarze. Dla sekwencji o większym ruchu lepszymi algorytmami są E3SS oraz „E3SS mod.1". Wyniki otrzymane dla algorytmu E3SS oraz algorytmu „E3SS mod.1" są zbliżone. Zatem z punktu widzenia implementacji sprzętowej, lepszym algorytmem jest zmodyfikowany algorytm E3SS z ogra-

niczoną liczbą etapów wymaganych do wyznaczenia najlepszego dopasowania. Maksymalny czas potrzebny do przetworzenia danych jest równy czasowi trzech etapów działania zmodyfikowanego algorytmu.

Algorytm z ograniczeniem etapów przetwarzania cechuje się zbliżoną wartością szczytowego stosunku sygnału do szumu PSNR, do wartości PSNR oryginalnego algorytmu E3SS. Algorytm „E3SS mod.2" lepiej radzi sobie z wyznaczaniem najlepszego dopasowania w przypadku scen z małym ruchem, natomiast gorsze wartości PSNR otrzymywane są dla scen z większym ruchem. Jednak średni szczytowy stosunek sygnału do szumu dla tak zmodyfikowanego algorytmu zrównuje się ze średnim PSNR algorytmu E3SS przy sekwencjach zawierających większą liczbę ramek (rys. 3).



Rys. 3. Wykres zależności średniego szczytowego stosunku sygnału do szumu w zależności od liczby ramek zawartych w sekwencji testowej *tennis*

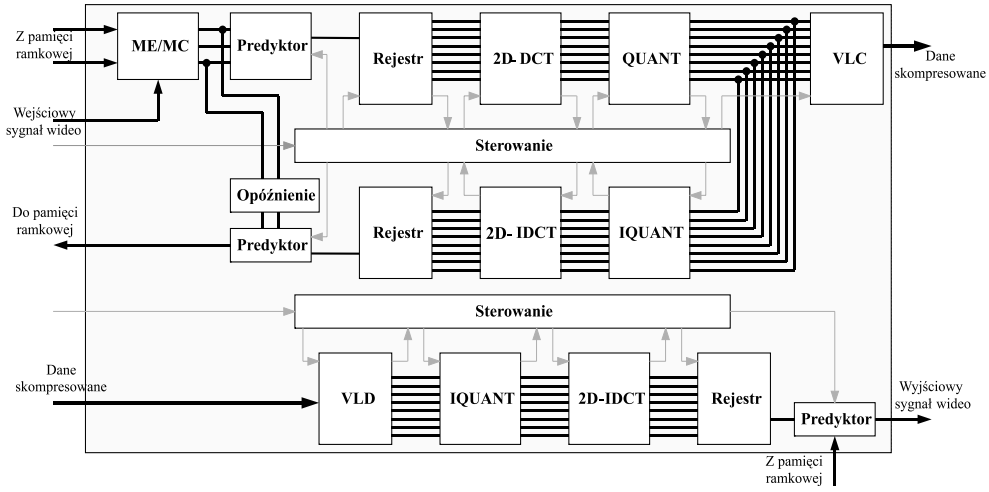
5. Sprzętowa implementacja standardu MPEG-2

Pierwszym zagadnieniem, które należało rozwiązać w przypadku implementacji kodera oraz kodeka standardu MPEG-2, była taka implementacja bloków dyskretnej transformacji kosinusowej DCT oraz odwrotnej transformacji kosinusowej IDCT, aby bloki te mogły przetwarzać dane wejściowe w sposób ciągły. Rozwiązaniem było zastosowanie dwóch

oddzielnych bloków 2D-DCT pracujących naprzemiennie. Jednak takie rozwiązanie wymagało dwóch bloków pamięci transpozycji oraz czterech bloków odpowiedzialnych za 1D-DCT, z których w danym momencie tylko dwa bloki były efektywnie wykorzystywane. Lepszym sposobem okazała się realizacja wykorzystująca tylko dwa bloki odpowiedzialne za 1D-DCT pracujące w sposób ciągły oraz pamięć transpozycji wykorzystującą, tak jak w poprzednim rozwiązaniu, dwa bloki pamięci rozproszonej – pracujących naprzemiennie.

Kolejnym aspektem, który należało rozpatrzyć, było zapewnienie ciągłości pracy bloku estymacji ruchu, gdyż pierwotnie czas przetwarzania pojedynczego makrobloku był równy sumie czasów potrzebnych do zapisania pamięci przechowującej obszar przeszukiwania oraz działania samego procesu wyznaczania najlepszego dopasowania. Rozwiązaniem tego problemu było zastosowanie pięciu bloków pamięci, z których każdy odpowiedzialny jest za inny obszar przeszukiwania. W danym momencie, tylko jeden z tak zaimplementowanych bloków pamięci jest źródłem danych dla działającego bloku algorytmu estymacji ruchu.

Wszystkie działania związane ze zrównolegleniem obliczeń oraz związane z samą optymalizacją kodu VHDL miały na celu zmniejszenie zajmowanych zasobów sprzętowych układu FPGA, wykorzystywanych przez zaimplementowany koder/kodek przy jednoczesnym dużym priorytecie na szybkość przetwarzania danych wejściowych.



Rys. 4. Schemat blokowy kodeka standardu MPEG-2 wykorzystującego obrazy I, P oraz B

Zaimplementowany w układzie XC4VFX100-11FF1152 kodek standardu MPEG-2, wykorzystujący wszystkie typy obrazów, wymaga 19977 bloków SLICE – co stanowi 47% zasobów tego układu (rys. 4). Na taką zajętość składa się 13245 przerzutników Flip-Flop, 932 zatrzaski oraz 27785 bloków LUT (32% z 84352), z czego 24313 bloków wykorzystuje logika, 887 bloków zajmuje routing, 2560 zajmuje rozproszona dwuportowa pamięć RAM. Dodatkowo wykorzystanych jest 178 ze 376 dostępnych bloków BRAM oraz jeden układ

DCM – zarządzający zegarem układu. Przy typowej budowie grupy obrazów IBBPBBPB-BIBB przy zastosowaniu procesu estymacji ruchu, kodek jest w stanie przetwarzać obraz bez przeplotu o rozdzielczości 1920×1152 pikseli 8-bitowych i 26 ramek na sekundę. Kodek może jednocześnie pracować w trybie kodera i dekodera. Jeśli ważnym parametrem będzie liczba ramek na sekundę, kodek może pracować w trybie z „zerowym" wektorem ruchu. Można wtedy osiągnąć szybkość 67 ramek na sekundę przy rozdzielczości HDTV, jednak konsekwencją tego jest mniejszy współczynnik kompresji. Dodatkowym atutem opisanego rozwiązania jest to, że łączy w sobie zarówno koder jak i dekoder, które mogą pracować niezależnie.

6. Podsumowanie

Pierwszym elementem, który został przystosowany do implementacji sprzętowej w układach FPGA pod kątem zajętości zasobów oraz szybkości działania algorytmu, była dyskretna transformacja kosinusowa DCT. Spośród kilku możliwych algorytmów DCT w wyniku badań jako algorytm najlepiej nadający się do sprzętowej implementacji został wytypowany algorytm Chena. Modyfikacja tego algorytmu, dokonana przez autorów pracy, polegała na ograniczeniu dokładności odwzorowania współczynników mnożenia, przy czym nie wszystkie współczynniki zostały zaimplementowane z taką samą dokładnością. Współczynniki mnożenia odpowiedzialne za wyznaczanie współczynników DCT niosących ze sobą największą energię, zostały zaimplementowane z większą dokładnością. Natomiast współczynniki mnożenia odpowiedzialne za wyznaczanie pozostałych współczynników zostały odwzorowane z mniejszą dokładnością. Działanie takie powodowało powstawanie różnicy pomiędzy pikselami wejściowymi a ich rekonstrukcją otrzymaną poprzez złożenie transformacji 2D-DCT i 2D-IDCT. Jednak maksymalna otrzymana różnica jest prawie niezauważalna dla oka ludzkiego. Ponadto wykazane zostało, że maksymalna różnica pomiędzy współczynnikami otrzymanymi z zaimplementowanego zmodyfikowanego algorytmu Chena a współczynnikami wyznaczonymi z podstawowej zależności opisującej 2D-DCT jest mniejsza niż najmniejszy możliwy współczynnik macierzy kwantyzacji. Sytuacja taka skutkuje wystąpieniem możliwego błędu na najmłodszym bicie skwantowanej wartości współczynnika dotyczącego składowej stałej. Oznacza to, że po procesie dekwantyzacji otrzymany współczynnik dotyczący składowej stałej może różnić się maksymalnie o ±8. Mimo takiej różnicy pomiędzy wyjściem z 2D-DCT a wejściem 2D-IDCT, zrekonstruowane piksele nie różnią się od swoich pierwowzorów o więcej niż 3.

W przypadku implementacji funkcji celu, decydującej o wyznaczaniu najlepszego dopasowania dla makrobloków, z obrazu obecnie przetwarzanego, wśród makrobloków obrazów referencyjnych, została uwzględniona wartość maksymalnej różnicy między pikselem a jego rekonstrukcją. Znalazło to swoje odzwierciedlenie przy wyborze funkcji celu decydującej o wyznaczaniu wektorów ruchu. Jako funkcja, która najlepiej nadaje się do sprzętowej implementacji, została wybrana funkcja klasyfikacji różnicy pikseli. Zaletą tej

funkcji jest możliwość sterowania dokładnością wyznaczania najlepszego dopasowania poprzez wartość progową. W przypadku takiej implementacji 2D-DCT wartość progowa nie powinna być mniejsza niż $|\pm 3|$.

W wyniku badań została opracowana implementacja kodeka standardu MPEG-2, który wykorzystuje wszystkie typy obrazu: I, P oraz B. Kodek taki jest w stanie kompresować w czasie rzeczywistym obraz o rozdzielczości HDTV (np. 1920×1152) przy użyciu procesu estymacji i kompensacji ruchu. Dodatkowo możliwy jest tryb kodowania z „zerowym" wektorem ruchu.

Literatura

[1] ISO/IEC 13818-2: 1995 MPEG video standard, ITU-T H.262 Recommendation.

[2] Netravali A.N., Haskell B.G., *Digital Pictures. Representation, Compression and Standards*. New York, Plenum Press 1995.

[3] Rabbani M., Jones P.W., *Digital Image Compression Techniques*. Bellingham, SPIE Optical Engineering Press 1991.

[4] Skarbek W. (red.), *Multimedia. Algorytmy i standardy kompresji*. Warszawa, Akademicka Oficyna Wydaw. PLJ 1998.

[5] Sayood K., *Kompresja danych – wprowadzenie*. Warszawa, Wyd. RM 2002.

[6] Domański M., *Zaawansowane techniki kompresji obrazów i sekwencji wizyjnych*. Wydawnictwo Politechniki Poznańskiej 2000.

[7] Heim K., *Metody kompresji danych*. Warszawa, Wyd. MIKOM 2000.

[8] Bhaskaran V., Konstantinides K., *Image and Video Compression Standards*. *Algorithms and Architectures*. 2nd Edition. Boston/Dordrecht/London, Kluwer Academic Publishers 1997.

[9] Symes P.D., *Video compression: Fundamental compression techniques and an overview of the JPEG and MPEG compression systems*. New York, McGraw-Hill 1998.

[10] www.xilinx.com.

[11] Furht B., Greenberg J., Westwater R., *Motion estimation algorithms for video compression.* London, Kluwer Academic Publishers 1997.

[12] Lap-Pui Chau, Xuan Jing, *Efficient three-step search algorithm for block motion estimation in video coding*. ICASSP '03, vol. 3, 421–424.

[13] http://bmrc.berkeley.edu/ftp/pub/multimedia/mpeg/EncodeData/tennis.tar.gz.