AUTOMATYKA • 2009 • Tom 13 • Zeszyt 3

Sławomir Cichoń*

Koncepcja implementacji trzyetapowego dekodowania VLC (w standardzie cyfrowego wideo DV) w układzie FPGA

1. Wprowadzenie

Kodowanie VLC (*Variable Length Coding*) jest odmianą kodowania entropijnego. W różnych odmianach występuje ono w wielu standardach kompresji, zarówno obrazów ruchomych, jak i nieruchomych, m.in. JPEG, MPEG-2, IEC 61834-2 (DV – *Digital Video*). Kodowaniu w powyższym standardzie poddawane są współczynniki otrzymane w rezultacie dwuwymiarowej transformaty kosinusowej 2D-DCT i kwantyzacji. Jeden lub kilka kolejnych współczynników w bloku DCT koduje się do jednego słowa kodowego o zmiennej długości. Słowa kodowe są zazwyczaj stabelaryzowane dla danego standardu kompresji. Jest to operacja, w której dokonuje się właściwa i bezstratna kompresja danych. Ciągom współczynników, które mogą występować najczęściej, przypisuje się najkrótsze słowa kodowe, natomiast ciągom współczynników, mogącym wystąpić bardzo rzadko, przypisuje się najdłuższe słowa kodowe. W pracy opisano algorytm kodowania VLC stosowany w kompresji DV, oraz zaproponowano koncepcję implementacji dekodera VLC w układzie reprogramowalnym. Częściowo algorytm ten opisano w pracach [1] i [4]. W pracy [3] zaprezentowano implementację pierwszego etapu dekodowania VLC w układzie FPGA dla standardu DV. Przykłady sprzętowych architektur dekoderów VLC dla kompresji MPEG-2 przedstawiono w pracach [5] i [6].

2. Kodowanie VLC w standardzie DV (IEC 61834-2)

Kodowanie VLC w standardzie DV jest operacją przekształcania ciągu wejściowego, jakim są zmodyfikowane współczynniki AC uzyskane po procesie kwantyzacji. Jeden lub kilka kolejnych współczynników AC w bloku DCT koduje się do jednego słowa kodowego o zmiennej długości.

* Motorola Polska Electronics, Sp. z o.o., slawomir.cichon@agh.edu.pl

Długość ciągu i amplituda (C, A) określone są w następujący sposób:

(C, A): para długości ciągu i amplitudy;

C: liczba kolejnych współczynników AC, których wartość po kwantyzacji wynosi 0 (C = 0, ..., 61);

A: wartość bezwzględna współczynnika następującego bezpośrednio po kolejnych współczynnikach AC skwantowanych do 0 (A = 0, ..., 255).

Każdej parze (C, A) w procesie kodowania przyporządkowywane jest słowo kodowe.

Tabela 1 przedstawia długości słów kodowych odpowiadających parom (C, A). Nie uwzględnia ona bitu znaku dla niezerowego współczynnika AC. Jeżeli amplituda nie jest zerowa należy długość kodu zwiększyć o 1.

Tabela 1

Długość słów kodowych w bitach w kodowaniu VLC

Długość ciągu	Amplituda																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	.....	255
0	11	2	3	4	4	5	5	6	6	7	7	7	8	8	8	8	8	8	9	9	9	9	9	15	.....	15
1	11	4	5	7	7	8	8	8	9	10	10	10	11	11	11	12	12	12								
2	12	5	7	8	9	9	10	12	12	12	12	12														
3	12	6	8	9	10	10	11	12																		
4	12	6	8	9	11	12																				
5	12	7	9	10																						
6	13	7	9	11																						
7	13	8	12	12																						
8	13	8	12	12																						
9	13	8	12																							
10	13	8	12																							
11	13	9																								
12	13	9																								
13	13	9																								
14	13	9																								
15	13																									
⋮	⋮																									
61	13																									

UWAGI
1. Bez bitu znaku
2. Długość EOB = 4

Fragment tablicy słów kodowych pokazano w tabeli 2. Najstarszy bit następnego słowa kodowego jest najbliższy najmłodszemu bitowi poprzedniego słowa kodowego. Bit znaku „s" powinien przyjmować jedną z dwóch wartości:

- s = 0, dla współczynnika AC większego od zera;
- s = 1, dla współczynnika AC mniejszego od zera.

Jeżeli wartości wszystkich współczynników AC pozostałych w bloku DCT po kwantyzacji są zerowe, proces kodowania bloku kończy się przez dodanie po ostatnim słowie kodowym, słowa kodowego 0110_2 EOB (*End Of Block*), oznaczającego koniec bloku DCT.

Pełną tablicę kodową zawiera pozycja [2].

Tabela 2
Fragment tablicy kodowej

(ciag, amp)		Kod			Długość
7	2	111110110000 s			12+1
8	2	111110110001 s			
9	2	111110110010 s			
10	2	111110110011 s			
7	3	111110110100 s			
8	3	111110110101 s			
4	5	111110110110 s			
3	7	111110110111 s			
2	7	111110111000 s			
2	8	111110111001 s			
2	9	111110111010 s			
2	10	111110111011 s			
2	11	111110111100 s			
1	15	111110111101 s			
1	16	111110111110 s			
1	17	111110111111 s			
6	0	1111110000110			13
7	0	1111110000111			
R	0	1111110	R w notacji dwójkowej R = 6 do 61		
61	0	1111110111101			
0	23	111111100010111 s			15+1
0	24	111111100011000 s			
0	A	1111111	A w notacji dwójkowej R = 23 do 255		
0	255	111111111111111 s			

UWAGI
1. ‘s’ jest bitem znaku

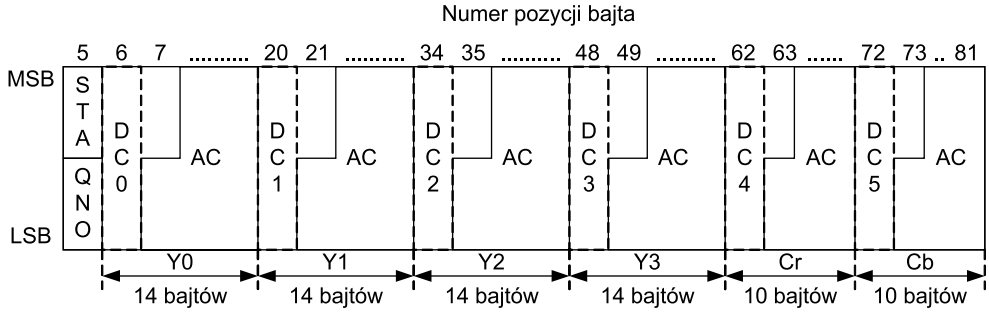
2. (R,0) 1111110 r5 r4 r3 r2 r1 r0
gdzie 32r5 + 16r4 + 8r3 + 4r2 + 2r1 + r0 = R

3. (0,A) 1111111 a7 a6 a5 a4 a3 a2 a1 a0
gdzie 128a7 + 64a6 + 32a5 + 16 a4 + 8a3 + 4a2 + 2a1 + a0 = A

3. Rozmieszczenie zakodowanych danych w segmencie wizji

Kodowanie VLC w standardzie DV jest operacją przekształcania ciągu wejściowego, jakim są zmodyfikowane współczynniki AC uzyskane po procesie kwantyzacji. Jeden lub kilka kolejnych współczynników AC w bloku DCT koduje się do jednego słowa kodowego o zmiennej długości. Fragmenty ramki wideo pogrupowane są w tzw. segmenty wizji. Każdy z nich składa się z 5 makrobloków, wybranych wg deterministycznego schematu, z róż-

nych fragmentów ramki. Schemat reprezentuje fragment ramki o rozmiarze 8×8 pikseli. Z kolei każdy makroblok składa się z 6 bloków DCT: 4 bloków luminancji (Y) oraz po jednym bloku chrominancji czerwonej (Cr) i niebieskiej (Cb). Wynika to ze stosowanego próbkowania 4:1:1. Ilość pamięci przeznaczona na skompresowane dane segmentu wizji jest stała i wynosi 400 bajtów (5×80 bajtów). Struktura skompresowanego makrobloku przedstawiona jest na rysunku 1.



Rys. 1. Struktura skompresowanego makrobloku

Dane uzyskane w wyniku kodowania VLC dla każdego z bloków DCT, a tym samym makrobloków, nie mają ustalonej długości. W celu optymalnego wypełnienia obszaru pamięci przeznaczonego dla skompresowanych danych segmentu wizji, ich rozmieszczanie przebiega w trzech etapach (przebiegach).

Algorytm rozmieszczania skompresowanych danych dla jednego makrobloku można zapisać w postaci pseudokodu przedstawionego na rysunku 2.

4. Koncepcja implementacji 3-etapowego dekodera VLC w układzie reprogramowalnym

Dekodowanie VLC jest najbardziej skomplikowaną i zarazem wymagającą największej ilości zasobów logicznych operacją, z punktu widzenia implementacji w układzie FPGA. Implementacja pierwszego etapu dekodowania VLC na potrzeby dekodowania sygnału wideo została zrealizowana w architekturze potokowej i opisana w pracy [3]. Przeprowadzone badania jakości dekodera DV realizującego algorytm dekodujący tylko 1 etap VLC wykazały, że współczynnik PSNR określający jakość przetwarzanego obrazu utrzymywał się na poziomie > 40 dB, względem tej samej sekwencji zdekodowanej komercyjnym 3-etapowym dekoderem programowym. Wyniki uznano za dobre i bardzo dobre. Wskazano jednak na możliwość poprawy jakości dekodowania w przypadku zaimplementowania algorytmu 3-etapowego.

```
AC2P = 0; /* AC2P jest ciągiem bitów dla danych nieprzydzielonych w
             etapie 2 */

/* etap 1 - rozmieszczenie słów kodowych AC w wyznaczonych strefach
wewnątrz makrobloku*/
for(i=0;i<5;i++) /* po makroblokach w segmencie  wizji */
{
    AC1P = 0; /* AC1P jest ciągiem bitów nieprzydzielonych do
    makrobloku   w etapie 1 */
       for(j=0;j<6;j++) /* po blokach DCT */
       {
              ciag_nieprzydzielony = rozmiesc(VLC[i][j], MB[i][j]);
        AC1P = polacz(AC1P, ciag_nieprzydzielony);
    }
}

/* etap 2 - rozmieszczenie nieprzydzielonych słów (lub/i ich
fragmentów) w strefach innych bloków DCT      w obrębie tego samego
makrobloku*/
for(i=0;i<5;i++)
{
       for(j=0;j<6;j++)
              AC1P = rozmiesc(AC1P, MB[i][j]);
       AC2P = polacz(AC2P, AC1P);
}

/* etap 3 - rozmieszczenie słów kodowych (także ich fragmentów)
pozostałych po etapie 2 w strefach AC danego segmentu wizji */
for(i=0;i<5;i++)
{
       for(j=0;j<6;j++)
              AC2P = rozmiesc(AC2P, MB[i][j]);
}

/*************************************************************/

/* Przydziela dane 'ciag_bitow' od najstarszego bitu, począwszy od
pierwszego wolego bitu strefy 'obszar' */
/*  'pozostaly_ciag' jest ciągiem bitów, który nie został
przydzielony */
rozmiesc(ciag_bitow, obszar)
{
       pozostaly_ciag = nieprzydzielone dane;
       return pozostaly_ciag;
}

/* Dołącza (konkatenuje) ciąg bitów 'dane2', do najmłodszego bitu
'dane1' */
polacz(dane1, dane2)
{
    dane3 = sklej(dane1,dane2);
       return dane3;
```

Rys. 2. Algorytm kodowania VLC w standardzie DV

Algorytm 3-etapowy jest jednak znacznie trudniejszy do zaimplementowania w zasobach FPGA. W szczególności trudność polega na opracowaniu architektury umożliwiającej prawidłowe buforowanie fragmentów słów kodowych odpowiadające za drugi i trzeci etap, rozproszone w obrębie segmentu wizji. Realizacja dekodera pracującego w czasie rzeczywistym, w systemie strumieniowym opisana w pracy [3], zakłada buforowanie informacji wizyjnej dla dwóch makrobloków. Pełne dekodowanie VLC, łącznie z drugim i trzecim etapem, wymaga buforowania co najmniej jednego segmentu wizji. Zmienia się tym samym granulacja obliczeń. Wprowadzony jest kwant danych o większej ziarnistości w postaci segmentu wizji.

W prezentowanej koncepcji przed dokonaniem właściwego dekodowania scalono rozproszone skompresowane dane poszczególnych bloków DCT i zapisano je w buforach pamięci. Dopiero stamtąd dane pobierane są w celu dekodowania. Schemat blokowy dekodera, przedstawiono na rysunku 3.

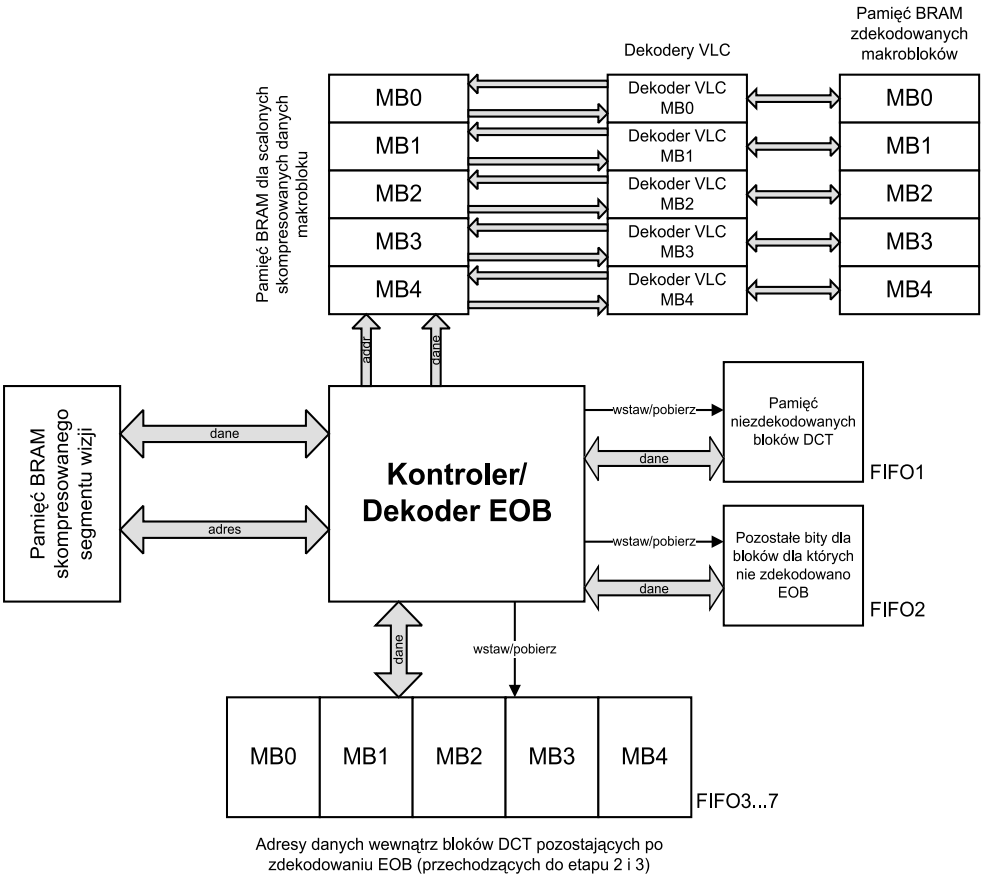
Dane wejściowe segmentu wizji, dla etapu dekodowania VLC, zapisywane są w pamięci blokowej BlockRAM. Jest to dedykowany zasób w układzie FPGA, w postaci bloków pamięci dwuportowej o dostępie swobodnym, rozmieszczonych po całej strukturze układu programowalnego. Rozmiar tej pamięci to 400 bajtów (80 bajtów dla każdego z makrobloków). Głównym blokiem całego dekodera jest Kontroler/Dekoder EOB, który spełnia kilka zadań:

- pobiera dane z pamięci skompresowanego segmentu wizji;
- po pobraniu danych dekoduje zgodnie z tabelą 1, w poszukiwaniu słowa kodowego EOB;
- dla danego bloku DCT, w przypadku osiągnięcia końca obszaru danych (14 bajtów dla bloków Y, 10 bajtów dla bloków CR, CB) i nie zdekodowania słowa EOB, zapisuje w kolejce FIFO1: indeks dekodowanego bloku DCT (0…5), oraz indeks makrobloku (0…4). Do kolejki FIFO2 zapisywany jest ciąg bitów niezdekodowanych i dla danego bloku DCT, oraz jego długość (maks. 16 bitów);
- dla danego bloku DCT, w przypadku zdekodowania symbolu końca bloku EOB, do odpowiedniej kolejki FIFO (FIFO3…FIFO7) zapisywane są adresy danych wewnątrz bloków DCT, pozostających po zdekodowaniu symbolu EOB;
- kopiuje słowa kodowe z pamięci wspólnej dla segmentu wizji, do obszaru pamięci BRAM przeznaczonego dla scalonych danych w etapie 2 i 3.

Dla każdego bloku DCT, który nie został w pełni zdekodowany w danym etapie, w kolejce FIFO1 zapisywany jest jego indeks w obrębie makrobloku (0...5) oraz numer makrobloku (0…4), do którego przynależy. Dane z tej kolejki służą układowi kontrolera/dekodera do wyznaczenia adresu wewnątrz pamięci blokowej, przechowującej scalone skompresowane dane segmentu wizji. Głębokość kolejki FIFO1 jest równa liczbie bloków DCT w segmencie wizji, czyli 30 (6 bloków DCT × 5 makrobloków).

Kolejka FIFO2 służy do zapisania, dla każdego bloku, dla którego nie zdekodowano słowa EOB, pozostałego ciągu bitów w ramach obszaru danych dla danego bloku DCT.

W najgorszym wypadku jest to 15 bitów, ze względu na to, że długość najdłuższego słowa kodowego wynosi 16 bitów. Głębokość tej kolejki, podobnie jak FIFO1, wynosi 30. Uwzględniony jest w ten sposób warunek graniczny, w którym w pierwszym etapie dekodowania, nie udało się znaleźć słowa EOB dla żadnego z bloków w segmencie wizji.



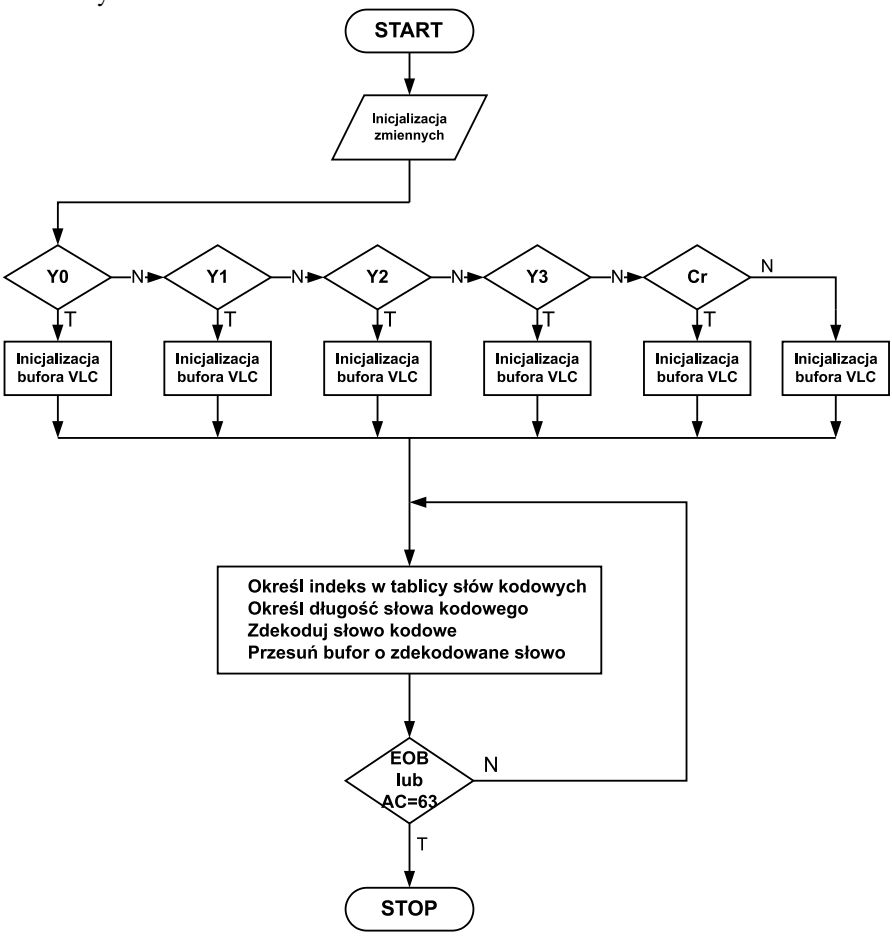
Rys. 3. Schemat ideowy 3-etapowego dekodera VLC

W kolejkach FIFO3…FIFO7 zapisywane są, dla właściwego indeksu makrobloku, podczas dekodowania, adresy z przestrzeni adresowej pamięci wejściowej segmentu wizji, danych, które przechodzą do drugiego i/lub trzeciego etapu dekodowania. Zapisany adres powinien określać początek danych z dokładnością do pojedynczego bitu. Głębokość tych kolejek powinna być równa liczbie bloków DCT w obrębie makrobloku, czyli powinna być równa 6.

Dane wyjściowe w postaci scalonych słów kodowych dla poszczególnych bloków DCT zapisywane są w pamięci blokowej układu FPGA. Ponieważ długość scalonego ciągu

słów kodowych może być różna, w zależności od dynamiki i detali zarejestrowanej sceny, dlatego jako rozmiar tej pamięci przyjęto 150% rozmiaru pamięci wejściowej segmentu wizji.

Sam proces dekodowania scalonych danych polega na porównaniu odpowiedniej liczby bitów słowa kodowego z wzorcem wyróżniającym pewną grupę słów kodowych, a następnie, w zależności od zdekodowanego słowa, zapisanie w odpowiedniej komórce pamięci blokowej bloku DCT zdekodowanego współczynnika. Algorytmiczny opis tego etapu przedstawia rysunek 4.



Rys. 4. Dekodowanie pojedynczego makrobloku, dane makrobloku scalone

5. Wnioski

Dekoder VLC stanowi pierwszy etap w dekompresji sygnału DV. Etap ten w różnych modyfikacjach i z różnymi postaciami tablicy słów kodowych obecny jest w niemal wszyst-

kich standardach kompresji sygnału wideo. Dekodowanie polega na przekształceniu wejściowego strumienia bitowego w postaci sklejonych ze sobą dobrze zdefiniowanych słów kodowych na ciąg 64 współczynników, które poddane dalszym etapom dekompresji utworzą fragment zdekompresowanej ramki obrazu. Z punktu widzenia implementacji w układzie FPGA jest to najbardziej skomplikowana i zarazem wymagająca największej ilości zasobów logicznych operacja. Zaprezentowana architektura dekodera VLC pozwala na zrealizowanie wszystkich trzech etapów dekodowania, przy jednoczesnym ograniczeniu ilości bloków funkcjonalnych, oraz bloków pamięci dedykowanych wyłącznie dla jednego z etapów dekodowania. Zaproponowana architektura pozwala na rozbudowę dekodera VLC do postaci umożliwiającej dekodowanie wszystkich etapów, w strukturze potokowej kompletnego dekodera DV [3], a przez to uzyskanie poprawy jakości dekodowanego obrazu.

Literatura

[1] *DVCAM Format Overview*. Sony Corporation, 2000.
[2] CENELEC, Norma europejska EN 61834-2, *Format SD dla systemów 525-60 i 625-60*, październik 1998.
[3] Gorgoń M., Cichoń S., Pac M., *Real-time Handel-C based implementation of DV decoder*. Proceedings 2005 International Conference on Field Programmable Logic and Applications (FPL), 130–135, IEEE 05EX1155, IEEE, Piscataway, New York, USA, 2000.
[4] Kwiecień P., *Implementacja w układach FPGA modułów sprzętowych obsługujących cyfrową transmisję obrazu*. Katedra Automatyki AGH, Kraków, 2001.
[5] Min K., Chong J., *A memory efficient VLC decoder architecture from MPEG-2 application*. IEEE Workshop on Signal Processing Systems SiPS, Lafayette, USA, 2000, 43–49.
[6] Qu Y., Mei S., *A Cost-effective VLD Architecture for MPEG-2 and AVS*. Journal of Signal Processing Systems, vol. 52, Kluwer Academic Publishers, Hingham, USA, 2008, 95–109.