2001년 한국정보처리학회 추계 학술발표논문집 제8권 제2호

웹 카메라 서버용 JPEG2000 의 EBCOT 설계에 관한 연구

박 주현*, 김 영철*, 홍 성훈*, 이 명옥**
*전남대학교 전자공학과
**(주)하이칩스 와 동신대학교 전자정보통신공학부

e-mail : jhpark@neuron.chonnam.ac.kr

Design of an EBCOT in JPEG2000 for a Web Camera Server

Ju-Hyun Park*, Young-Chul Kim*, Sung-Hoon Hong*, Myung-Ok Lee**
*Dept of Electronic Engineering , Chonnam National University
**Hi-Chips Co.Ltd and Dept of Electrical and Information and comm. Engineering , Dongsin University

요 약

본 연구는 웹 카메라에 적용하기 위한 JPEG2000의 주요 블록인 EBCOT(Embedded Block Coding with Optimized Truncation)의 설계 및 구현에 관한 연구이다. EBCOT 블록은 웨이블렛(wavelet)변환에 의해 분할된 각 sub-band에 존재하는 주위 화소 값들과 상위 bit-plane의 값들에 대한 상호 연관성을 조사하여 context을 추출하고 그 context를 이용하여 엔트로피 부호화(entropy coding)를 수행하는 T1(Tier 1) 블록과 bit-stream을 구성하는 T2(Tier 2) 블록으로 구성된다. 본 논문에서는 JPEG2000에서 전체 압축성능을 좌우하는 EBCOT의 T1 블록을 Synopsys tool을 이용하여 설계하고 구현하였다.

1. 서 론

현재 정지영상 압축 부호화 방식으로 사용되고 있는 JPEG의 단점으로는 저 비트율(0.25 bpp 이하)에서 발생하는 블록화 현상으로 인한 화질 열화와 무손실 압축과 손실압축이 통일되어 있지 않다. 또 손실 JPEG (IS 10918-1)과 무손실 JPEG-LS (IS 14495-1)가 따로 있으며 압축해제 알고리즘이 여러 가지 이다.

이러한 JPEG 압축 성능의 한계를 극복하여 wavelet coding 기술의 비약적인 발전과 새로운 압축형식의 경쟁, 그리고 의료 및 전자 상거래 등의 다양한 응용 분야에 대한 적용을 목적으로 기존 JPEG을 지원하면서, 향상된 압축 효율 및 다양한 기능성을 제공하는 새로운 형태의 정지영상 압축의 표준인 JPEG2000의 필요성이 크게 대두되었다.

JPEG2000의 목표는 20대 1정도인 기존 정지화상 압축률을 최소한 5배 이상 향상된 1백대 1이상으로 고 밀도 화하고, 기존 JPEG 적용 시 한계인 텍스트, 컴퓨터 그래픽, X레이 및 인공위성 사진 등 모든 종류의 정지화상을 고 밀도로 압축해 화질에 손색없이 정지화상으로 구성하는 것이다.

이러한 JPEG2000 은 크게 Wavelet 블록, 양자화 블록, EBCOT 블록으로 나눌 수 있다.

EBCOT 의 특징은 메모리 요구가 줄어들며 각 블록이 독립적으로 coding 되기 때문에 병행 성이 증가하고 블록별 통계에 따른 최소 왜곡으로 압축 성능을 높일 수 있기 때문에 본 논문은 EBCOT를 구현하므로 압축처리 시간을 향상시키고 더 나아가 JPEG2000 의 구현을 목적으로 한다.

본 논문의 구성은 제 2장에서 EBCOT에서 coding 종류와 context 추출 방법을 설명하고 제 3장에서는 산술 부호화기에 대해 설명하며 제 4장에서는 설계 및 구현에 대해 설명한다. 마지막으로 제 5장에서는 본 논문에 대한 결론 및 향후 연구 계획에 대해 설명한다..

2. EBCOT 의 coding 과 context 추출 방법

EBCOT는 subband sample 값들을 이용하여 entropy coding을 수행(T1)하는 것뿐만 아니라 그 결과 code를

1)본 논문은 시스템 직접 반도체 기반 기술개발 사업 연구비에 의하여 연구되었으며 IDEC 지원 설계 tool을 사용하였음.

bit-stream에서 packing 하고 구성(T2) 한다.
EBCOT 구성은 그림 1, 그림2 와 같다.

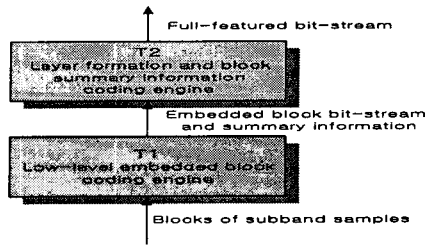

그림 1. EBCOT 구성

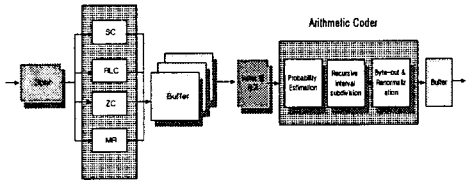

그림 2. T1 의 구성 블록

EBCOT coding system은 양자화 된 결과를 bit-plane 형태로 나누고 각 bit-plane 별로 entropy coding을 위해 context를 추출한다.
Context 추출을 위해서 상위 bit-plane 의 정보뿐 아니라 주위 pixel 값들의 정보를 사용한다.
Context 추출을 위한 coding은 4가지가 있다.

2.1 ZC(Zero Coding)

Significance state가 false(Insignificance)이고 즉, 상위 bit-plane에서 아직까지 '1'이 발견되지 않은 상태의 sample에 대해 현재 bit-plane에서 유무(1/0)를 부호화한다. 현재의 sample값을 부호화하기 위해서는 주위 8개의 sample값과 sub-band의 특성을 참조로 하여 만들어진 표 1을 참조하여 context 하나를 선택하게된다. Context는 모두 9가지(0~8)가 제시되어 있으며 실제로 arithmetic coding table에서는 label 0~8을 의미한다.

표 1. ZC로 추출되는 context 와 h,v,d 조건

LH sub-band(also used for LL) (vertically high-pass)				HL sub-band (horizontally high-pass)				HH sub-band (diagonally high-pass)		
h	v	d	context	h	v	d	context	d	h+v	context
2	×	×	8	×	2	×	8	≥3	×	8
1	≥1	×	7	≥1	1	×	7	2	≥1	7
1	0	≥1	6	0	1	≥1	6	2	0	6
1	0	0	5	0	1	0	5	1	≥2	5
0	2	×	4	2	0	×	4	1	1	4
0	1	×	3	1	0	×	3	1	0	3
0	0	≥2	2	0	0	≥2	2	0	≥2	2
0	0	1	1	0	0	1	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0

2.2 RLC(Run Length Coding)

ZC 결과를 이용하여 RLC를 실행하는데 RLC를 실행하는 조건은 다음과 같다.

- 4개의 연속된 표본들이 모두 insignificance
- 4개의 모든 표본들이 동일하게 zero neighborhood (ZC neighborhood 문맥 변수 ,h, v, d = 0)
- 모든 4개의 표본들이 수평적으로 인접하고 같은 sub-block 내에 위치

위의 조건을 만족하는 그룹이 있을 때, 그 그룹 내에 있는 sample들 중에서 현재 bit-plane 에서 significant sample이 있는지 부호화한다. 이 때 context는 하나뿐이며, 위의 조건을 만족하고 4개의 모든 sample이 insignificant 이면 uniform context를 실행한다.
4개의 연속된 sample에 대해 significant가 발견되면 2-bit을 할당하여 위치 정보를 보낸다.

2.3 SC(Sign Coding)

ZC, RLC 중에 significant symbol이 발견되면 수행한다. 각각의 code-block sample에 대해 한번만 수행된다. horizontal, vertical neighbor, sign predictor를 이용하여 현재 위치의 context(5개)를 추출하며 실제로 arithmetic coding table에서 index값은 9~13이다. Horizontal , vertical 내에서의 significance symbol에 관하여 coding 하기 때문에 방향성은 무시된다. 따라서 h, v 값은 -1, 0, 1값을 갖게 된다.

표 2. SC로 추출되는 context 와 h, v 조건

h	v	x	context
1	1	0	4
1	0	0	3
1	-1	0	2
0	1	0	1
0	0	0	0
0	-1	1	1
-1	1	1	2
-1	0	1	3
-1	-1	1	4

2.4 MR(Magnitude Refinement)

Significance state 가 true인 sample에 대해서 현재 bit-plane에 해당하는 비트를 부호화하는데 여기서 h, v는 zero coding에서 h, v와 동일하며, 상위 bit-plane 에서 MR이 적용되었는지 새로운 상태 변수를 사용한다. 이 상태 변수는 처음에 "0" 으로 초기화되고,
MR operation 이 완료되면 "1" 로 설정된다. 추출되는 context는 모두 3가지이며, 표 3 과 같다.
이는 arithmetic-coding table의 index에서 14~16에 해당된다.

표 3. MR로 추출되는 context 와 h, v 조건

$\delta_i[m,n]$	h+v	context
1	×	2
0	≥1	1
0	0	0

3. 산술 부호화기

산술 부호화기는 4가지 coding에 의해 추출된 context를 기반으로 entropy coding을 실행한다. 산술 부호화는 huffman-coding과는 달리 입력 symbol들에 대해 누적 분포를 산술 적으로 계산하여 code word를 생성하게 된다. 산술 부호화는 입력 symbol의 길이가 길어져도 쉽게 부호화 할 수 있다는 장점이 있다.

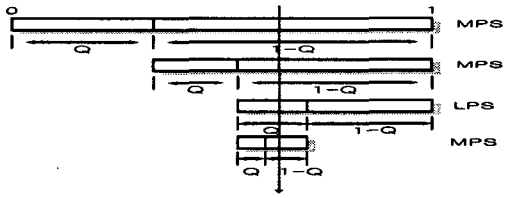

그림 3. 산술 부호화기의 Coding 과정

산술 부호화기의 누적 확률 구간은 [0,1] 사이로 주어지며 여기에서는 symbol의 표현을 0과 1대신 MPS ,LPS로 표현한다.

Coding 방법은 그림 3과 같이 MPS, LPS 구간의 선택에 따라 산술 적으로 계산을 실행하고 마지막 선택된 구간의 누적 분포 값들 중 가장 작은 값을 code word 로 생성하게 된다.

JPEG2000에서는 산술 부호화기를 기존에 사용되던 Q-Coder 나 QM-Coder의 곱셈과 덧셈을 사용하는 것 대신 MQ-Coder를 사용하였는데 이 MQ-coder에서는 덧셈과 shift를 사용하여 처리 시간과 성능을 개선하였다.

4. 설계 및 구현

EBCOT 구현을 위해 먼저 bit-plane 형태로 나누어주는 블록과 각각의 coding 블록들을 synopsys tool을 사용하여 설계하였다.

4.1 EBCOT 의 4가지 coding 블록

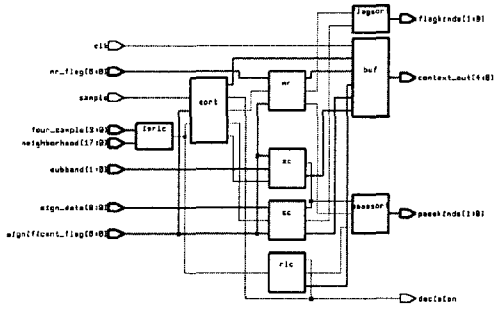

그림 4. EBCOT 4가지 coding블록의 세부 블록도

양자화 된 데이터를 받아들여 zero coding, sign coding, magnitude refinement, run_length_coding을 거쳐 context를 추출하는데 그 context는 arithmetic coder에 사용되어 code_word를 생성하게 된다.

그림 5는 4가지 coding 블록들의 simulation 결과이다.

시뮬레이션 결과를 살펴보면, 현재 simulation은 최상 위 bit-plane에 관한 코딩 결과이며, significant_flag, sign_flag는 모두 '0'을 갖는다. 4개의 연속된 샘플이 무효화 상태(significant_flag = '0')이고 주위 8개의 샘플들도 무효화 값을 갖기 때문에(significant_flag 값을 통해 알 수 있다.) RLC를 시작한다. 다음 clk에서는 4개의 연속 된 샘플의 값("1000")이 모두 zero 인지를 확인하는데, 위에서 들어오는 데이터는 첫 번째 데이터가 "1000" 인 non_zero (유효화 계수) 값을 갖는다.

따라서 유효 값의 위치를 uniform coding을 해주게 되는데, uniform을 나타내는 context "00010001"(11)을 출력하게 된다.

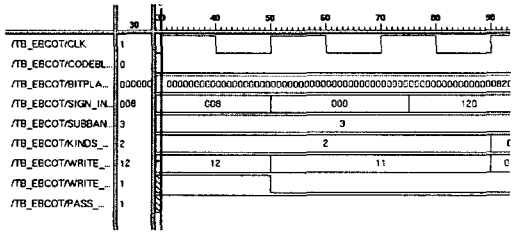

그림 5. EBCOT 4가지 coding블록 simulation 결과

4.2 산술 부호화기

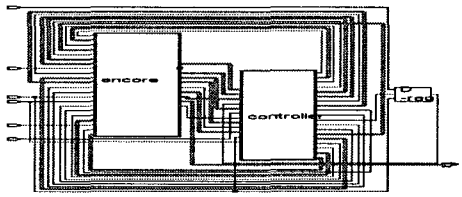

그림 6. 산술 부호화기의 세부 블록도

각 context 에 맞는 index의 설정 블록을 두어 산술 부호화기에 맞는 index를 입력으로 받아들이고, 그 index에 맞는 구간의 확률 값을 Rom에 미리 설정하였다. 또한 산술 부호화기 자체 control 블록을 두어 산술 연산을 제어하였다.

그림 7은 산술 부호화기 simulation결과인데, 살펴보면 simulation 결과에서 CT 가 "0" 이 되었을 때 그때 c-reg 의 값 "Ox0007EFE8" 결과 값에서 19 ~ 26 bit 값 "00000000"을 출력 한 것을 볼 수 있다. 출력 할 때의 Qe 값은 "0000101011000001" 이며 그때의 index 값은 30 이다. clk 은 10 ns 의 주기로 설정하고 c-reg (code register)와 a-reg(interval register) 초기 값들은 각각 c-reg = '0' ,a-reg = "1000000000000000" 으로 설정하였으며 CT (counter)는 12로 설정했다.

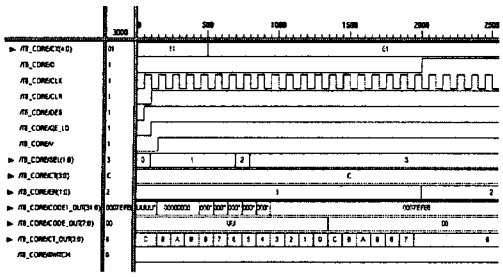

그림 7. 산술 부호화기의 simulation 결과

4.3 EBCOT top 블록

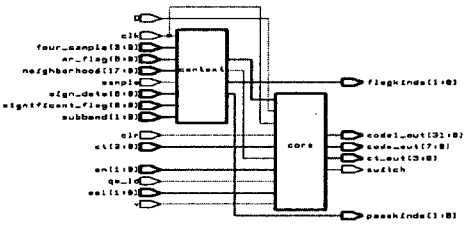

그림 8. EBCOT 전체 세부 블록도

그림8에서 4가지 coding(ZC, RLC, MR, SC)을 실행하는 블록은 context block 이고 산술 부호화기의 블록은 core block 이다.

그림 9는 전체 top simulation 결과인데, simulation 결과를 살펴보면 4개의 연속된 샘플이 '0'이고 주위 8개의 샘플들도 '0' 값을 갖기 때문에 RLC를 시작한다. 다음 4개의 sample 값 "1000"에서 '1'인 계수를 가지므로 그 위치를 uniform coding을 해주게 되고 MR-flag 값이 '1'로 변화되는걸 볼 수 있다.

또한 그때의 context 값인 "11"을 이용하여 산술 부호화기를 실행하였는데 c-reg 와 a-reg 초기 값들은 각각 c-reg = '0' ,a-reg = "1000000000000000", CT는 12로 설정하였고 CT 가 "0" 이 되었을 때 code 값인 "00000000"이 출력되는걸 볼 수 있다.

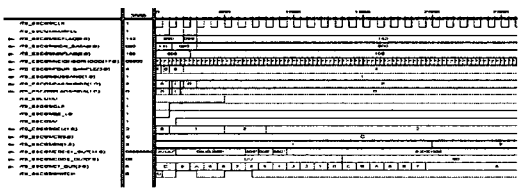

그림 9. EBCOT 전체의 simulation 결과

5. 결론

본 논문에서는 메모리 요구가 줄어들며 각 블록이 독립적으로 coding 되므로 병행 성이 증가하고 최소 왜곡으로 압축 성능을 높일 수 있는 EBCOT를 설계하고 구현하므로 압축 처리 시간을 향상시킬 수 있었다.

또한 산술 부호화기(MQ-coder)를 설계하므로 기존에 사용하던 산술 부호화기(Q-coder, QM-coder) 보다 연산량을 줄이고 처리 시간을 향상시킬 수 있었다.

향후 연구 과제로는 설계되어진 EBCOT를 기반으로 bit-stream을 구성하는 T2(tier2) 부분을 설계 및 구현하며 더 나아가 JPEG2000의 전체 top 블록을 설계하고 구현하는 것이다.

참고문헌

[1]Charilaos Christopoulos, "JPEG2000 verification model 8.5",
ISO/IEC JTC 1/SC 29/WG 1 WG1 N1878

[2]Charilaos Christopoulos, Martin Boliek, "JPEG2000 Part I Final Committee Draft Version 1.0",
ISO/IEC JTC 1/SC 29/WG 1 N1646R

[3]PAUL G.HOWARD, JEFFREY SCOTT VITTER, "Arithmetic Coding For Data Compression" ,IEEE

[4]M.J.Slattery, J.L.Mitchell IBM J. RES, "THE Qx-coder" DEVELOP. VOL 42..NO 6, November 1998