H.264 CAVLC(Context-Adaptive Variable Length Coding)설계

이용주 ** 서 기 범 **

*우송대학교

A design of CAVLC(Context-Adaptive Variable Length Coding) For H.264

Yong ju Lee** Ki-bum Suh**

*Electronic Dept. Graduate School, Woosong University

E-mail : yjlee81@wsu.ac.kr

요 약

본 논문에서는 동영상의 실시간 Full HD 영상(1920x1080@30fps) 부호화를 위한 하드웨어 기반의 CAVLC 엔트로피 부호화기 구조를 제안한다. 한 매크로 블록 당 AC 계수 376개 와 DC 계수 8개 총 384개의 데이터가 존재 할 수 있다. 실시간으로 처리하기 위해서는 최대 384개의 데이터를 모두 처리해야 한다. 데이터를 효율 적으로 처리하기 위해 병렬 처리, 파이프라인 처리를 사용, 블록당 16개의 데이터 이후의 존재하는 불필요한 '0' 제거로 동작 cycle를 최소화 하였다. 설계된 모듈은 한 매크로 블록당 최대의 384개의 데이터를 469cycle로 처리한다. CAVLC 구조를 검증하기위하여 JM 9.4부터 reference C를 개발하였으며, reference C로부터 test vector를 추출하여 설계 된 회로를 검증하였다

ABSTRACT

In this paper, we propose an advanced hardware architecture for the CAVLC entropy encoder engine for real time Full HD video compression. Since there are 384 data coefficients which are sum of 376 AC coefficient and 8 DC coefficient per one macroblock, 384 coefficient have to be processed per one macroblock in worst case for real time processing. We propose an novel architecture which includes parallel architecture and pipeline processing, and reduction "0" in AC/DC coefficient table. To verify the proposed architecture, we develop the reference C for CAVLC and verified the designed circuit with the test vector from reference C code.

키워드

H.264, CAVLC ,Entropy coding

Ⅰ. 서 론

H.264는 비디오 부호화에 해당하는 기능과 저장매체에 저장하거나 통신망을 통한 전송을 위한 기능을 구분하기 위하여 부호화기의 구조를 비디오 부호화 계층(Video Coding Layer)과 네트워크 적응 계층(Network Abstraction Layer)으로 분리하였다. 이런 특징으로 H.264는 회선 기반 망이나 패킷 기반 망을 통하여 효율적으로 데이터를 전송할 수 있도록 NAL Unit(Network Abstraction Layer Unit)이라는 구조화된 형태로 미디어 데이터를 저장한다[1].

본 논문에서는 베이스 프로파일 레벨 3규격의 H.264 코덱용 ASIC의 CAVLC 모듈의 하드웨어를 설계하였다. 그림 1과 같이 부호화 과정의 최종 단계는 Entropy Coding이다. Entropy Coding은 심볼의 출현 빈도에 의한 확률에 근거하여 최적의 코드워드를 배정하는 통계학적 예측에 기초하고 있다. 본 논문에서는 모든 심볼들을 UVLC(Universal VLC) 테이블로 처리하는 부분과 4x4 양자화 된 계수 처리에 특화되어 설계된 CAVLC(Context-Based Adaptive Variable Length Coding)로 나누어 설계 하였다.

제안된 CAVLC모듈의 기능 및 성능에 관련된 주요 특징은 NAL Unit 과 Annex B.1의 스트림 생성을 지원하며 매크로블록보다 상위의 syntax에 대한 비트스트림을 생성할 수 있다. 이때 매크로블록 상위정보는 AMBA 인터페이스를 통하여 공급된다.

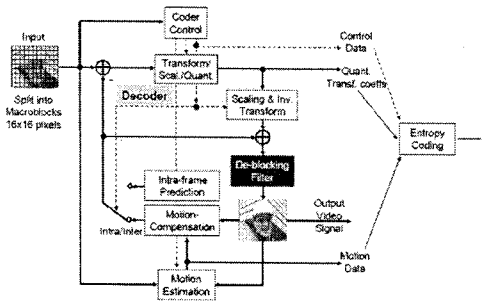

그림 1. H.264 Encoder 구조

또한 H.264/AVC 높은 효율과 다양한 전송 환경에서도 에러에 강한 특성을 지닌 최신의 영상 압축 표준화 방식이다. H.264/AVC 영상 압축 표준 방식이 위와 같은 높은 부호화 효율을 갖는 것은 다양한 크기의 움직임 보상 및 1/4 화소 단위의 움직임 추정, 인트라 추정, 정수 기반 DCT 변환, 디블락킹 필터와 엔트로피 부호화 같은 기술을 적용했기 때문이다. 기존의 H.263이나 MPEG4와 달리 새로운 기술을 채택함으로써 압축 효율은 2배 이상 향상되었지만, 처리해야 할 연산량은 증가했고 복잡도는 최대 16배 이상 증가했다. 따라서 실시간 Full HD급을 처리할 수 있어야 하는 시스템을 설계하기 위해 하드웨어 기반의 구조 설계가 요구 되고 있다.

II. CAVLC(Context-Based Adaptive Variable Length Coding)

H.264에서는 하나의 매크로블록 단위로 엔트로피 부호화가 수행된다. 하나의 매크로블록은 그림 2와 같이 4x4 블록이 25개, 2x2 블록이 2개의 블록으로 구성된다.

각 블록은 그림 2와 같이 미리 정의된 순서에 따라 부호화된다. 매크로블록의 부호화모드가 I16MB mode일 때는 4x4 Luma DC(Direct Current)블록(block -1)이 부호화되고, 4x4 Luma AC(Alternate Current) 블록(block 0-15)이 부호화된다. 그 다음으로 색도 DC(block 16-17)이 부호화 되고, 색도 AC(block 18-25)가 부호화 된다. 매크로 블록의 부호화 모드가 I16MB가 아닐 때는 4x4 Luma DC 블록의 부호화가 생략된다.

CAVLC는 지그재그 스캔된 4x4 block과 2x2 block의 변환 계수들의 오차 블록을 부호화하는 데 사용되는 방법이다. CAVLC는 양자화 된 4x4 block의 여러 가지 특징을 이용하기 위해 만들어 졌다.[3]

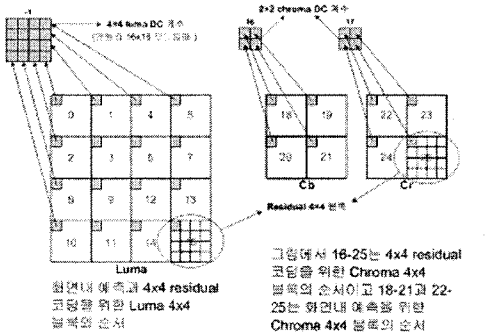

그림 2. Macroblock의 Encoding 순서

변환 계수 블록의 CAVLC 인코딩은 다음과 같은 순서로 진행된다.

1. 계수의 개수와 TrailingOnes를 부호화한다.
2. 각 TrailingOnes의 부호를 부호화한다.
3. 나머지 0이 아닌 계수들의 레벨을 부호화한다.
4. 마지막 계수 이전의 전체 0의 개수를 부호화한다.
5. 각 0의 run을 부호화한다.

III. 새로운 CAVLC 하드웨어 구조의 필요성

3.1 Rate control

H.264의 표준은 실시간 비디오 커뮤니케이션을 목표 디자인 되었다. 현대에는 네트워크 속도가 발전하여 고화질의 IP TV가 등장 하고 있다. H.264는 옵션으로 Rate Control를 지원하고 있다. IP TV에서 Rate Control를 사용한다면 매우 유리하다. 네트워크 상황에 따라 화질이 조절되면서 네트워크 상황에 맞게 스트림이 발생 된다면 이용자에게 더 좋은 화질과 쾌적한 영상시청이 가능하다. H.264의 Rate Control은 실시간으로 출력되는 스트림 량을 파악하고 QP값을 변화하면서 출력되는 스트림 량을 균형 있게 조절하는 역할을 하고 있다. 만약 복잡한 영상일 경우 스트림 량이 증가 할 것이다. 그 경우 H.264의 Rate control은 QP값을 올려 출력되는 스트림을 균형적으로 나오게 할 것이다. 다른 입장에서 영상이 단순했다면 출력 스트림이 작게 나올 것 이다. 그러므로 H.264의 Rate control은 QP을 내려 그 부분에선 고화질의 영상을 제공하면서 출력스트림을 균형적으로 유지 될 것이다. 그림 4는 akiyo cif 영상의 bit rate 4096 kbps에서의 QP의 변화량을 나타낸 그림이다. 8번째 프레임을 보면 QP

가 0까지 떨어지는 걸 볼 수 있다. 따라서 QP 0 ~ 51까지 어떤 상황이 오더라도 인코딩 해내야 한다. QP가 0에 가까워지면 순간 스트림 량이 증가 하게 될 것이다. 이 부분에서 CAVLC에서는 병목 현상 없이 스트림을 처리 하기위해 제안된 구조는 QP의 변화에 상관없이 일정한 클럭 내에 처리 할 수 있는 CAVLC encoder의 설계가 필수적 이다.

akiyo_cif_ldr15_rcOn_br4M_qp28.psnr

Frame	POC	Pic#	QP	SnrY	SnrU	SnrV
0000(I)	0	0	28	40.2564	42.4924	44.1712
0001(P)	2	1	28	40.0138	42.4401	44.2058
0002(P)	4	2	28	41.8266	43.3522	45.1655
0003(P)	6	3	22	44.5259	45.9882	47.3584
0004(P)	8	4	17	47.4752	48.3834	49.372
0005(P)	10	5	12	50.8954	51.2266	51.7865
0006(P)	12	6	7	54.9754	55.0212	55.1409
0007(P)	14	7	2	57.8327	57.8063	57.8624
0008(P)	16	8	0	57.8745	57.4846	57.6044
0009(P)	18	9	0	58.0939	57.8785	57.8559
0010(P)	20	10	0	58.3041	58.2163	58.232
0011(P)	22	11	0	58.2624	58.3117	58.2076
0012(P)	24	12	0	57.2932	57.2461	57.2644
0013(P)	26	13	0	57.0597	56.8842	57.0174

그림 3. Rate Control bit rate 4096 kbps

3.2 통합 Pipeline

기존의 제안된 구조들은 CAVLC의 평균 적인 Cycles 수를 나타내었는데 하지만 실제 인코더를 설계하게 되면 인코더의 각각의 중요 모듈을 파이프 기법으로 통합 될 것이다.

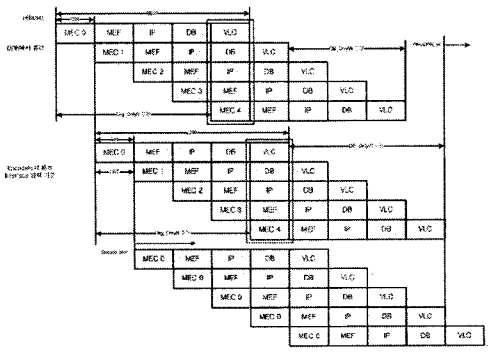

그림 4. 5단 Pipeline

위 그림 3번은 통합 인코더 모듈의 파이프라인 그림이다. 파이프라인 기법을 사용하여 인코더가 설계되었다면 평균 Cycles수가 중요한 게 아니고 worst case의 Cycles수가 중요하다. full HD급 영상을 인코딩하려면 MB수가 증가함에 따라 더 작은 Cycles수를 필요로 하게 된다. 그리고 그 파이프라인을 유지하기위해서는 파이프라인에 할당된 Cycles수를 지켜줘야 한다. 만약 할당된 Cycles수를 넘어 버린다면 파이프라인에 심각한 오류를 유발 할 것이다. CAVLC의 worst case에는 I4MB에 LUMA AC 계수 16개 X 16 블록 CHROMA DC 8개 CHROMA AC 15 X 8블록 총 384개의 데이터를 처리해야 한다. 그리고 MB 해더 부분 역시 처리 해주어야 한다. 위의 데이터를 처리 하려면 많은 Cycles를 소비 하게 될 것이다.

위 케이스를 논문[5]번과 같이 코딩한다면 초기 mb의 정보인toal_coeff, num_coeff, trailing_ones, total_zoro 를 스캔 하는 Cycles수는 최소 블록당 16 Cycles를 소비 하게 된다. 실제 코딩에서는 Coeff_token 1 Cycle, Sign Trail 1 Cycle, Levels 16 Cycle, Total Zoros 1 Cycle, Run_before 부분은 Level 와 병렬가능 패킹 1 Cycle 총 1블락당 16 + 1 + 1 + 16 + 1 + 1 = 36 Cycle를 요하게 된다. LUMA AC (36 x 16) + CHROMA AC (34 X 8) + CHROMA DC (12 X 2) = 872 Cycles 필요로 하게 된다. 위와 같은 worst case의 MB가 연속적으로 나오게 된다면 그 인코더는 심각한 오류를 발생 하게 될 것 이다.

제안하는 CAVLC 부호화기는 어떤 상황이 나오더라도 고정된 Cycles로 처리하는 구조 이다. 고정된 Cycles는 469 Cycles 이다. 제안하는 CAVLC는 MB syntax 와 MB layer를 고정된 469 Cycle안으로 처리가 된다.

Ⅳ. 제안된 CAVLC 부호화 기의 구조

그림 5는 본 논문에서 제안 하는 CAVLC 부호화기의 구조를 보여준다.

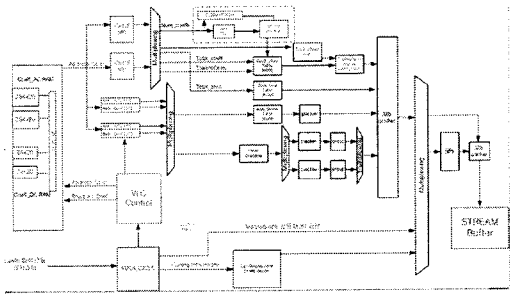

그림 5. CAVLC Encoder Architecture

위의 전체 블록도를 보면 AMBA CAVLC 모듈은 NAL Unit 과 Annex B.1의 스트림생성과 매크로블록보다 상위의 syntax에 대한 비트스트림, MB layer syntax를 코딩 하는 모듈이다. Coeff_AC 와 DC는 듀얼버퍼링 기법을 사용하였다. Packer모듈을 여러 개 사용하여 병렬적 패킹을 가능하게 하였다. FIFO모듈역시 2개를 사용하여 파이프라인 기법을 사용할 수 있게 한다.

4.1 Pipeline기법 사용

MB syntax 부분을 제외한 CAVLC의 구조에 따라 블록 코딩을 3단 파이프라인을 구성하였다.

1 : 초기 정보 2 : VLC 코딩 3 : 종합 패킹

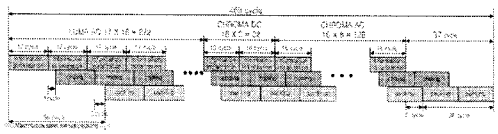

그림 6. CAVLC Pipeline

그림 6은 CAVLC에서 사용된 파이프라인 이다. 3단 파이프라인 이므로 초기 2단에서는 perking 이 일어나지 않는다. 그 부분에서 MB layer syntax를 코딩 하게 된다. 그리고 파이프라인은 LUMA(17cycle) 블록과 CHROMA(16cycle) 블록에 따라 사이클이 다르게 움직이도록 하여 사이클 수를 줄일 수 있도록 만들어져 있다. 이 파이프라인은 worst case 기준으로 설계되었으며 469 cycles 에 처리 가능하다. 위 경우는 I4mb의 경우이고 만약 I16MB 가된다면 첫 블록(LUMA DC)만이 17 cycle로 동작하고 나머지 블록은 16cycle로 동작한다. 이유는 I16MB 일 경우 AC의 개수가 최대 15가 나올 수 있기 때문이다. 그러므로 I16MB 모드에서의 worst case 일 경우에도 I4MB의 worst case와 동일한 469 cycles로 코딩 가능 하다.

4.2 모든 모듈의 병렬 패킹 지원

그림 6의 패킹 cycle은 16(Chroma)~17(Luma) 이다. 기존의 perker는 worst case기준 코딩시 4x4 블록 코딩시 coeff_token 1 cycle, sign trail 1 cycle, level 15 cycle total Zoros 1 cycle run before 15cycle 총 33 cycle를 소모 하게 된다. 하지만 제안된 CAVLC 에서는 모든 모듈의 병렬 패킹 지원하여 level 패킹을 담당하는 Packer와 run brfore Packer를 둠으로써 15개의 레벨이 나오더라도 32비트 단위로 패킹하기 때문에 15보다 적은 수의 cycle로 패킹 한다. run before 는 최대 32bit가 넘을 수 없으므로 1 cycle로 코딩 가능 하다. token 과 Sign Trail 역시 두 데이터를 합쳐 1cycle로 처리 한다.

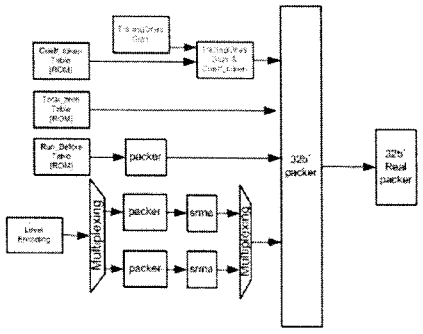

그림 7. 병렬 패킹

total zoros 1cyle 총 16cycle 안으로 코딩 가능 하다. 그림 7은 병렬 패킹을 수행 하는 모듈이다. 32비트 Packer는 파이프라인상 모든 코딩이 끝난 상태이기 때문에 병렬적으로 데이터를 받아 Real packer에 순차 적으로 패킹을 수행하는 모듈이다.

그림 8은 실제 modelsim 시뮬레이션 파형이다. 녹색으로 된 부분이 파이프라인으로 동작함을 보여주고 있다. 가장 앞 블록에서는 Packer 라인이 쉬는 간격을 이용하여 MB leyer syntax를 코딩하는 부분이다.

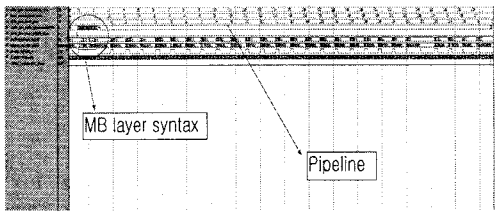

그림 8. modelsim 시뮬레이션 파형

IV. 결 론

본 논문에서는 Full HD급 영상을 실시간으로 처리 할 수 있는 효율적인 CAVLC하드웨어 설계 구조를 제안 하였다. 제안된 구조는 파이프라인 기법과 모든 모듈의 병렬 패킹 기법을 사용함으로써 어떠한 경우라도 안정되게 코딩할 수 있음을 확인 하였다. 제안된 하드웨어 구조는 Verilog HDL를 이용하여 설계 되었으며 JM 9.4[4]를 이용하여 검증 되었고, Full HD 1080 30fps의 영상을 115MHz에서 동작시킬 수 있음을 확인 하였다.

본 연구에 사용된 CAD Tool은 IDEC으로부터 지원 받았음.

참고문헌

[1] ITU-T Rec. H.264/ISO/IEC 11496-10, "Advanced Video Coding", Final Committee Draft, Document JVT-F100, October 2004

[2] High Performance Context Adaptive Variable Length Coding Encoder for MPEG-4 AVC/H.264 Video Coding Min-Chi Tsai and Tian-Sheuan Chang

[3] Iain E.G. Richardson "H.264 and MPEG-4 VIDEO COMPRESSION" John Wiley & Sons, Ltd. , 2003

[4] Joint Video Team reference software JM 9.4

[5] "A design of Contex-Based Adaptive Variable Length Coder For H.264" Hong-Sic lee