Motion JPEG용 허프만코딩 기반의 엔트로피 디코더 설계

김경현 · 손승일 · 이민수

한신대학교 정보통신학과

A Design of Huffman Coding Based on Entropy Decoder for Motion JPEG

Kyung-hyun Kim · Seung-il Sohn · Min-soo Lee

Dept. of Information and Communication HanShin University

E-mail : kkh52103@lycos.co.kr

요 약

정보화 사회가 진행되어감에 따라 카메라 센서, 디지털 카메라, 휴대폰, 영상 관련디지털 기기들이 증가하고 이로 인하여 영상정보 서비스 기술의 중요성이 크게 부각되었다. 특히 멀티미디어 응용서비스 기술에서는 영상 정보가 필수적인데, 그 영상 정보의 양이 너무 방대하여 압축 부호화를 하여 사용되고 있다. 본 논문에서는 정지영상압축 방법 중 JPEG표준에서 제시한 4가지 동작 모드 중 베이스라인을 기반으로 하는 JPEG 알고리즘을 연구하여 허프만코딩 기반의 엔트로피 디코더의 불규칙적인 데이터 입출력타이밍의 효과적인 제어를 통해 Motion JPEG에서 동작 가능한 디코더를 C언어를 통해 시뮬레이션하고 최적화된 결과를 VHDL로 구현하였다.

Ⅰ. 서 론

정보화 사회가 진행되어감에 따라 디지털 영상의 응용 분야가 급속하게 확대되고 있다. 이러한 응용분야 중 디지털 카메라의 ExIF(Exchangeable Image File Format)는 JPEG(Joint Photographic Experts Group) 압축 기술을 이용하는 업계 표준 파일 포맷이다. 이 표준의 가장 기본적인 기능들을 포함하는 기술인 JPEG 베이스라인 기술은 단순성과 폭 넓은 지지 기반 덕분에 디지털 카메라 시스템에 있어서 핵심기술이 되었다. 또한 JPEG을 연속적으로 인코딩하는 기법을 사용한 Motion JPEG기술을 통해 보안 분야 및 교통관련 영상 등의 카메라와 관련된 동영상 캡처기술이 널리 사용되고 있다[1].

JPEG의 인코더에서는 다량의 정보를 가진 영상신호 데이터 저장의 제약을 줄이기 위해 효율적인 영상 압축 기법을 적용 데이터양을 감소시켜 전송시킨다. 하지만 이에 따른 영상 신호의 복호화시 상대적으로 연산량 증가의 결과를 가져왔다. 이는 전력 소모 및 시간지연의 문제로써 나타나고 실시간 처리능력을 갖춘 저 전력 고속의 디코더를 위해서는 Motion JPEG 디코더 전용 하드웨어의 개발이 요구되었다[2].

이에 본 논문에서는 입력데이터의 효과적인 파이프라인 제어방식 및 고속의 연산 기법을 통해 하드웨어에 적합한 Motion JPEG용 허프만코딩 기반의 엔트로피 디코더를 연구하였다. 이를 토대로 하드웨어 설계언어인 VHDL을 이용하여 회로를 모델링하여 동작을 검증하였다.

Ⅱ. 엔트로피 디코더

2-1 불규칙한 데이터 입력

디코더 연구시 가장 우선적으로 고려해야 하는 사항은 입력데이터의 저장 방식이다. 프레임메모리 방식의 디코딩 장비는 특정 타이밍에 원하는 위치의 영상 데이터를 참조하여 쓸 수 있는 장점이 있지만 영상 데이터의 저장을 위한 메모리가 이미지 사이즈만큼 추가적으로 필요하다[3].

본 논문의 연구 주제인 Motion JPEG의 영상 디코딩 장비는 무선 스트리밍 방식으로 데이터를 통신하는 경우가 대부분이며 불규칙한 데이터 입력이 이루어 질 경우를 고려하여야 한다[2]. 이에 본 논문은 메모리 사용을 최소화하여 블록사이즈 크기 정도만 할당하여 연속적으로 데이터가 입력될 경우에도 데이터의 손실 없이 처리 가능하도록 하였다. 또한 데이터 전송 시 오류 및 손실에 의한 이미지 복원이 불가능할 때 해당 프레임의 디코딩을 중단하고 새로운 이미지정보를 바로 입력받아 처리함으로써 연속적인 이미지 처리의 오류의 누적을 막았다.

2-2 가변적인 데이터 길이 처리

엔트로피 디코딩 데이터의 입력 값은 Code word와 가변적인 유효 데이터 값이 연속적으로 연결되어 있는 형태로 입력된다. 이를 각각의 코드 정보와 고정비트의 압축 데이터로 복원하기 위해서는 한 번에 원하는 만큼 데이터를 가져올 수 있는 모듈이 필요하다[3][4]. 이에 본 논문에서는 그림1과 같이 Upperer / Lowerer 각각의 16 비트 레지스터와 코드길이 정보에 따른 쉬프트 길이 정보를 가산기를 통해 Barrel Shifter 모듈을 제어함으로써 불필요한 지연 없이 효율적으로 데이터를 추출하는 방식을 사용하였다.

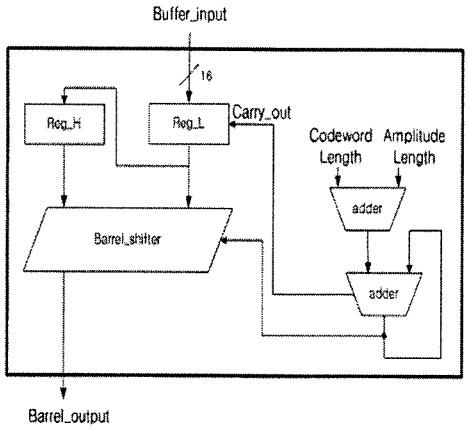

그림 1. Barrel Shifter

2-3 고속 AC 계수값 디코딩

2-3-1. AC코드 및 코드길이 생성기

각 코드길이별 계수 값은 식(1)을 통해 코드 길이가 k일 때 코드 내에서 r번째 계수 값 C를 계산으로써 얻을 수 있다[4][5].

$$C_k^r = \sum_{j=1}^{k-1} 2^{k-j} N_j + r, r = 0,1,\ldots,N_{k-1} \quad (1)$$

식(2)는 식(1)에 대한 코드길이(N)에 따른 최소 코드 값 C_K의 연산과정을 통해 유추하는 방식을 나타낸다.

$$\begin{aligned} c_2^0 &= 2N_1 = 0 [N_1 = 0] \\ c_3^0 &= 2c_2^0 + 2N_2 = 2(c_2^0 + N_2) = 2(0+2) = 4 \\ c_4^0 &= 2(c_3^0 + N_3) = 2(4+1) = 10 \end{aligned} \quad (2)$$

본 논문에서는 이러한 과정을 미리 수식으로 계산결과를 얻어 각 코드길이별로 최소 계수 값을 구하여 1비트단위 비교연산이 아닌 16비트 전체를 한 번에 비교함으로써 디코딩시 사용되는 불필요한 수행과정을 생략하였다.

2-3-2 허프만 코드 계수값 디코딩

JPEG Baseline에서 사용되는 기본적인 엔트로피방식에서는 허프만테이블을 이용한 코딩을 사용한다. 이때 사용되어진 테이블 정보를 모두 디코더에서 가지고 있기에는 메모리낭비가 커진다. 이에 표1은 AC테이블 정보 중 휘도(Luma)값의 예를 나타낸 표로써 헤더정보 Li값과 코드정보를 가지는 테이블 헤더를 ROM에 코드길이 별로 정리하여 전송되어 진다. DC값의 경우에는 Category정보를 AC값의 경우 Run/Size 정보를 통해 디코딩 할 수 있다[6][7][8].

표1. AC Luma Huffman Table 정보

Header Li	코드길이	ROM Data
00	1	·
02	2	01, 02
01	3	03
03	4	00, 04, 11
⋮	⋮	⋮
7d	16	09, 0A, 16, … FA

표1의 헤더정보 Li를 통해 코드길이별 각 허프만 테이블 정보를 ROM Data에 저장시키는 한 가지 예를 분석하면 만약 "FA"의 ROM Data가 선택되었다면 이는 이전의 '0'값의 개수를 나타내는 Run값이 'F'이고, 유효데이터의 데이터 비트 수를 나타내는 Size값이 'A'임을 나타낸다[7][8].

표2. AC Luma Huffman Table 정보

Luma(AC)			
Code length	Minimum code memory	M(C_k^{min})	Base Memory
2	0(00)	B	B
3	4(100)	B + 2	B - 2
4	10(1010)	B + 3	B - 7
5	26(11010)	B + 6	B - 20
6	58(111010)	B + 9	B - 49
7	120(1111000)	B + 11	B - 109
8	248(11111000)	B + 15	B - 233
9	502(111110110)	B + 18	B - 484
10	1014(1111110110)	B + 23	B - 991
11	2038(11111110110)	B + 28	B - 2010
12	4084(111111110100)	B + 32	B - 4052
13	X		
14	X		
15	32704(111111111000000)	B + 36	B - 32668
16	65410(1111111110000010)	B + 37	B - 65373

표3. AC Chroma Huffman Table 정보

Chroma(AC)			
Code length	Minimum code memory	M(C_k^{min})	Base Memory
2	0(00)	B	B
3	4(100)	B + 2	B – 2
4	10(1010)	B + 3	B – 7
5	24(11000)	B + 5	B – 19
6	56(111000)	B + 9	B – 47
7	120(1111000)	B + 13	B – 107
8	246(11110110)	B + 16	B – 230
9	500(111110100)	B + 20	B – 480
10	1014(1111110110)	B + 27	B – 987
11	2038(11111110110)	B + 32	B – 2006
12	4084(111111110100)	B + 36	B – 4048
13	X		
14	16352(11111111100000)	B + 40	B – 16312
15	32706(111111111000010)	B + 41	B – 32665
16	65416(1111111110001000)	B + 43	B – 65373

식(1),(2)를 통해 AC 계수 값을 디코딩시 ROM 주소를 얻어진 값을 정리하면 표2의 Luma Huffman Table과 표3의 Chroma Huffman Table과 같이 나타낼 수 있다. DC 테이블 값 디코딩시 테이블의 개수에 비하여 유추연산방식이 복잡하다고 판단 본 연구에서는 DC값은 코드 내부적으로 코드 값과 Category 값을 각각 매칭 하여 사용하였다[6][7][8].

III. 엔트로피 디코더 설계

3-1 엔트로피 디코더 전체 블록도

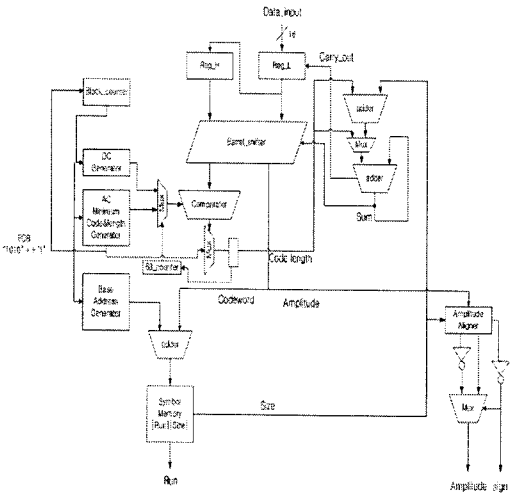

그림 2. 엔트로피 디코더 전체 블록도

그림2는 허프만테이블 기반의 Motion JPEG 디코더의 전체 블록도 이다. 최초 비트단위의 데이터들이 전송되어 처리 가능한 데이터 단위로 묶인 뒤 16비트 레지스터 Reg_L에 입력이 되고 이를 제어하는 가산기를 통해 Reg_L의 값은 16비트 단위로 Reg_H로 인가되는 형식으로 최악의 경우 Reg_H의 하위 0번째 비트와 Reg_L의 상위 15비트가 연결되어 Barrel Shifter에 입력이 될 수 있도록 한다. 이후 Block Counter 모듈의 제어에 따라서 Luma DC, Chroma DC, Luma AC, Chroma AC Generator 중 한가지의 비교기가 선택되어 동작하도록 한다. 비교기는 미리 계산된 Minimum code에 따라서 16비트 전체를 한 번에 조건 비교하여 바로 처리할 수 있도록 설계되었다. 이에 DC값의 경우 Code Length 와 Category 값을 유추하여 Code word와 Amplitude 값을 각각 쉬프터로부터 얻어낸다. AC값의 경우 추가적으로 Base Address Generator모듈의 연산을 통해 Run/Size 와 Code Length값을 유추해 낸 뒤 쉬프터로부터 유효한 데이터를 얻어낸다. 최종적으로 Run값은 별도 출력을 통해 이후 AC값의 데이터가 0인 부분을 패딩 시키며 그 외의 크기를 갖는 유효 데이터는 기존 1의 보수형태에서 2의 보수형태로 변환시켜 출력시켜 역양자화 모듈의 입력으로 사용된다.

3-2 시뮬레이션 결과

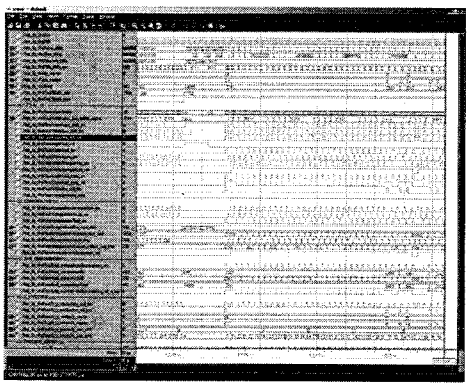

그림 3. 엔트로피 디코더 결과파형

IV. 설계 결과

표 4. 엔트로피 디코더 설계 결과

분 류	내 용
FPGA	Xilinx xc3s1000 fg676
Tools	Xilinx ISE 7.1i
Total Equivalent gate	7,477
Maximum Frequency	85.288MHz

설계된 Motion JPEG용 허프만 코딩 기반의 엔트로피 디코더 모듈은 표4와 같은 설계결과를 얻었다. XilinxISE 7.1i 개발 툴을 기반으로 Xilinx XC3s1000-4fg676환경에서 시뮬레이션 한 결과 최대 동작속도 85.288MHz를 나타내었고 총 7,477 게이트가 사용되었다.

그림 4. 320x240 복원영상

엔트로피 디코딩 이후 8x8 블록 단위로 역양자화 및 역변환 과정을 C언어를 통해 시뮬레이션 하여 입력데이터를 복호화 한 결과 그림 4와 같은 320x240 사이즈의 영상 결과를 얻었다.

V. 결 론

본 논문의 입력데이터는 선행 연구되어진 JPEG 인코더의 출력 값을 통해 전달하였고, 엔트로피 디코더의 오류검증은 우선적으로 기 설계된 엔트로피 인코더의 입력 데이터와 비교를 통해 이루어졌다. 이후 C언어를 통해 엔트로피 디코딩 이후 복호화에 필요한 처리 블록을 소프트웨어 모델링함으로써 영상으로써 검증하였다.

최종적인 모듈의 실장 테스트 결과 엔트로피 디코더는 하나의 계수 값을 복호화 하는데 3클록이 소요되었고 8x8 사이즈의 블록데이터 중 평균적으로 5개 이내의 화소만 유효한 데이터를 갖고 있었다. 이 유효데이터는 85MHz의 동작속도로 처리되었으며 이후 크기를 갖지 않은 연속적인 '0' 값의 데이터는 별도의 디코딩 작업 없이 Run 값 만큼의 카운터를 통한 패딩작업 만으로 전송시켰다. 그 결과 연속적인 초당 15프레임 수준으로 전송되는 Motion JPEG의 영상데이터 복호화 작업 수행이 가능하다.

향후 역양자화 및 역변환 모듈의 추가적인 연구 및 설계를 통해 디코더 블록을 완성시켜 완벽한 JPEG 코덱 IP 및 I^2C 인터페이스 모듈의 연결을 통해 카메라에서 입력받은 연속적인 영상을 영상장비의 디스플레이 장치에 실시간으로 복호화 시키는 모든 단계의 설계기술을 독자적으로 정립하고자 한다.

참고문헌

[1] 후지와라 히로시, 정제창 역, "최신 MPEG", 교보문고 1995.

[2] 박기현, "코덱의 세계로의 초대", 2007.

[3] "TMS320DM6446 Digital Media System-on-Chip," http://www.ti.com

[4] ITU-CCITT, "Information Technology Digital Compression and Coding of Continuous-Tone Still Images Requirements and Guideline", CCITT, 1993.

[5] Arun N. Netravali, Barry G. Haskell, "Digital Pictures", PLENUM Press, 1994.

[6] James Rosenthal, "JPEG Image Compression Using an FPGA, December 2006.

[7] Mohammed Elbadri, Raymond Peterkin, voicu Groza, Dan Ionescu, and Abdulmotaleb El Saddik "HADWARE SURPPORT OF JPEG", IEEE, 1995.

[8] Mario Kovac, N.Ranganathan, "A High Speed VLSI Chip for JPEG Image Compression Standard", IEEE, 1995.