Journal of Digital Contents Society Vol. 6. No. 3. September. 2005(pp. 169-174)

FPGA를 이용한 JPEG Image Display Board 설계 및 구현

권 병 헌* 서 범 석**

요 약

본 논문은 Verilog HDL로 FPGA에 JPEG Decoder를 구현하여 TV에 JPEG 영상을 디스플레이 하기 위한 JPEG Image Display Board 설계 방법을 제안한다. 본 논문은 FPAG에 JPEG Decoder Algorithm을 구현하기 위한 효율적인 방안을 제시하였으며 JPEG Decoding Algorithm은 JPEG Standard Baseline에 기준으로 하여 설계 하였다. 압축된 JPEG bit stream을 저장하기 위하여 Nand Flash Memory를 사용하였으며, JPEG Decoding된 영상을 TV화면에서 확인하기 위하여 Video Encoder를 사용하였다. 또한 JPEG 영상에 Text data를 쓰기 위하여 YCbCr의 출력 bit를 RGB 24bit로 변환하였다. Video Encoder에 변환된 RGB Data를 동기시켜 출력하기 위하여 CVBS 입력을 Sync Separator에 의해 Hsync, Vsync, Sync, Field signal로 분리하였다. 또한 Display B/D상의 스위치를 통하여 JPEG모드와 일반영상 모드를 선택할 수 있게 입증하였다.

Design and Implementation of JPEG Image Display Board Using FPGA

Byong-Heon Kwon*, Burm-Suk Seo**

Abstract

In this paper we propose efficient design and implementation of JPEG image display board that can display JPEG image on TV. we used NAND Flash Memory to save the compressed JPEG bit stream and video encoder to display the decoded JPEG image on TV. Also we convert YCbCr to RGB to super impose character on JPEG image. The designed B/D is implemented using FPGA.

Key words : FPGA, JPEG Image Display Board, DPCM, RLC

1. 서 론

데이터의 통계학적인 특성을 이용한 엔트로피 코딩은 영상처리, 음성처리 및 데이터 저장 등의 다양한 응용에 무손실 압축 기법으로 광범위하게적용되고 있다.

엔트로피 코딩 기술은 RLC (Run Length Coding), VLC (Variable Length Coding), arithmetic coding[1], Lempel-Ziv[2] 등에 의하여 구현되다. 앞의 3가지 코딩 방법은 비디오나 정지 화상의 표준 기술로서 사용되고 있다. 각 알고리즘의 성능은 원시자료의 통계성에 매우 의존하는 특성을 나타낸다.

본 논문에서는 정지 화상의 복원에 표준으로 채택되어 가장 일반적으로 사용되는 허프만 디코더 설계를 목표로 하였으며 허프만 코덱의 기본적인 알고리즘은 VLC 및 RLC 알고리즘으로 구성되어 있다.

정지 화상에 대한 표준은 ITU의 JPEG 권고안[3]을 기준으로 하고 이를 처리하기 위한 허프만 디코더 모델을 설정하였다. JPEG 또는 MPEG, HDTV에서 사용되는 허프만 코덱은 압축률을 보다 높이기 위하여 modified 허프만 모델을 사용하고 있다. 이 방법에서는 심볼의 수가 많아지면 허프만 트리의 수가 길어지게 되는 것을 방지하기 위하여 지리적으로 가까운 심볼들을 몇 개의 그룹으로 나누고 각 그룹의 위치에 해당하는 값을 허프만 부호화하고 그룹 내에서 각 심볼의 위치에 대한 정보를 부가 비트로서 보내게 되므로 트리가 상당히 길어지는 것을 방지할 수 있게 된다.

※ 제일저자(First Author) : 권병헌
접수일 : 2005년 7 월 10 일, 완료일 : 2005년 7 월 26 일
* 유학대학 정보통신공학과 부교수
** (주)플렛디스 책임연구원

2. 비디오 코덱 구조

대부분의 비디오 엔코더 및 디코더는 (그림 1)과 같은 DCT 기반의 JPEG용 엔코더 및 디코더에 대한 구조를 포함하고 있다. 엔코더에서 화소 데이터는 8x8의 2차원 영상을 하나의 블록 단위로 입력하여 DCT, 양자화 및 허프만 엔코더를 거쳐 압축 데이터를 발생한다.

DCT (Discrete Cosine Transform)은 손실이 있는 압축 알고리즘으로서 입력 2차원 영상을 1개의 DC 및 63개의 AC 계수로 변화한다. FDCT(Forward DCT)에서 입력은 9비트로 표현되는 부호있는 정수를 받고 IDCT (Inverse DCT)는 12비트의 부호 정수를 복원된 영상 데이터로 출력한다.

양자화 단계는 실수 계수인 DCT 계수를 유한한 개수의 양자화 값으로 나누어 정수 형태로 바꾸면서 계수의 크기를 줄여서 실제적인 압축이 일어나게 하지만 원영상에 손실을 가져오게 한다. 양자화 단계에 따라 압축 효율이 크게 영향을 받으며 양자화 계수를 크게 사용할수록 압축률은 높아지지만 화질의 열화를 크게 가져올 수 있다.

엔트로피 엔코더에서는 심볼의 확률에 따라 허프만 트리를 형성하고 이에 대한 허프만 코드를 형성한다. 압축된 영상 데이터는 비트열로 형성되어지고 전송 경로를 통하여 전송된다.

디코더는 엔코더에 대한 가역 순서로 처리된다. 엔트로피 디코더에서는 비트열로 전송된 입력 정보로부터 허프만 부호에 대한 심볼의 정보로 복호화하고 각 블록의 계수에 대한 역양자화 단계를 거쳐 IDCT를 실행하게 된다.

그림 2의 양자화 테이블은 양자화 단계에서 사용되는 양자화 테이블 값이다. JPEG에서 양자화 테이블은 규정치(default)가 따로 존재하지는 않으나 사람의 시각적 특성을 고려하여 typical하게 사용되는 값을 휘도 신호(Luminance) 및 색차 신호(Chrominance)에 대하여 정하고 있다. 양자화 테이블은 1-255의 값을 가지고 있는데, 사람의 시각을 고려하여 저주파 계수에 비하여 고주파 계수를 휘도 신호보다 색차 신호에 보다 더 큰 양자화 간격을 부여하고 있다.

허프만 테이블은 엔코더에서 허프만 트리를 형성할 때 심볼에 부과되는 코드를 저장하여두는 곳이다.

JPEG에는 규정된 허프만 테이블은 없으며 호상 데이터의 입력시 허프만 테이블을 생성하고 이를 함께 전송하여 복원시에 이를 참조하게 된다.

JPEG에서는 또한 영상에 대한 평균 통계적 값을 고려하여 작성된 typical 허프만 테이블을 함께 제공하고 있다[12]. Typical 허프만 테이블은 DC 및 AC에 대하여 각각 휘도 및 색차 신호에 대한 코드값을 가지므로 최대 4종류의 테이블이 존재한다[4].

본 논문에서는 modified 허프만 코딩과 이에 대한 JPEG의 후도 신호에 대한 typical 허프만 테이블을 사용하여 디코더 구조를 설계한다. 다른 부호를 사용하는 경우, 이에 대한 허프만 디코더의 구조는 동일하고 뒷절에서 설명하게 될 허프만 테이블을 저장하는 ROM 또는 PLA의 내용을 변화하면 된다.

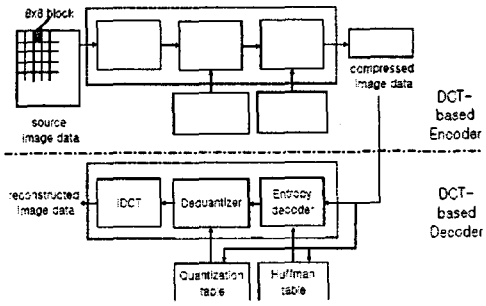

그림 1. DCT기반의 JPEG 코덱 구조

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	55	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

(a) Luminance

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

(b) Chrominance

그림 2. typical 양자화 테이블

3. JPEG Decoder 설계

엔트로피 디코더의 구조적인 설계 단계에 대하여 설명한다. 엔트로피 디코더의 단계는 허프만 코드를 먼저 복화화한 후 RS값으로부터 부가 비트에 대한 계수 값을 환산하고, Run length에 대한 zero를 삽입하는 과정을 수행하도록 하였다.

3.1 Huffman Decoder 모듈

압축된 데이터 스트림을 입력으로 하여 허프만 디코더는 복호화된 심볼의 값을 찾는다. 이 데이터 스트림은 가변 길이의 허프만 부호정보와 부가 비트로 구성되어 있기 때문에 코드로부터 한 개의 심볼에

대한 복호 정보를 찾고 이로부터 부가 비트에 대한 비트수를 환산한 후 비트 수만큼 입력으로부터 받은 계수 데이터를 그대로 출력하여야 한다. 허프만 부호에 대한 코드 테이블은 고정 테이블인 경우, ROM 또는 PLA를 사용한다. 코드의 입력에 대한 출력이 sparse한 성질을 가지고 있기 때문에 면적을 줄이기 위하여 PLA를 사용하여 구성하는 것이 유리하다. 부호의 내용이 가변적인 경우는 PLA대신에 RAM이나 CAM(Content Addre ssable Memory)을 사용할 수 있다. (그림 3)은 구현한 허프만 디코더의 구조도이다. 가변 길이의 입력 데이터는 입력 FIFO 구조를 사용하여 16비트로 저장한 후 두 개의 16비트 래치에 16비트 크기의 워드 단위로 저장한다[5].

Barrel shifter는 높은 자리 래치 입력을 받아서 16비트씩 PLA에 연결하고 코드 심볼이 매칭될 때 미리 저장된 RS와 코드 길이는 PLA의 출력으로 나타난다. Accumulator는 코드 길이를 누적하고 이 코드 길이 만큼 Barrel Shifter의 내용은 이동되면서 새로운 데이터가 래치로부터 채워진다. Accumulator의 내용이 16을 넘는 경우 발생된 캐리는 상하 래치의 내용을 갱신시킨다. (그림 3)의 구조 중 PLA는 DC 및 AC가 서로 다른 허프만 테이블을 사용하기 때문에 각 계수에 대하여 필요하다. 1개의 DC와 63개의 AC계수에 대한 디코딩된 정보는 차례로 공통의 출력으로 내보내어진다[6].

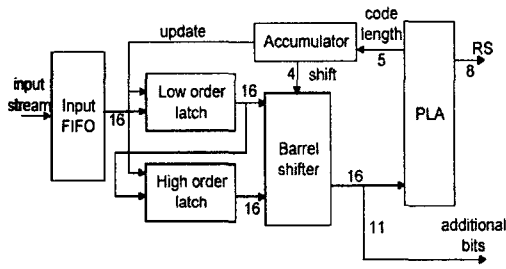

그림 3. Huffman Decoder 구조도

3.2 DPCM(Differential Pulse Code Modulation) 및 RLC(Run Legth Coding) 복원 모듈

Entropy Encoding의 동작에서 modified Huffman Model을 사용하여 Huffman Tree를 구성하는 경우, 디코더에서는 허프만 테이블로부터 RS정보와 코드 길이에 대한 정보를 얻고 부가 비트에 대한 정보를 함께 복호화된 정보로 출력하게 된다. DC계수에 대한 코드는 DPCM을 한 값에 대한 정보이므로 각 블록의 DC계수가 부가 비트로 나타날 때마다 이전 값에 더하여야 한다. DIFF에 대한 값은 2의 보수를 사용하여 표현되고 복원된 DC값은 양수만으로 나타나게 되어 최대 12비트로 표현 가능하다[7].

화상 frame의 첫번째 블록에 대한 DC값은 그대로 전송이 되어 이를 레지스터에 저장하여두고 이후 매 블록의 DC부가 비트를 이것과 더하면 된다. (그림 4)는 DPCM 모듈에 대한 구조도이다.

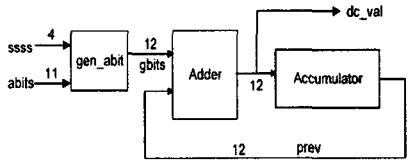

그림 4. DPCM 모듈 구조도

초기 DC값은 덧셈기인 Adder를 통하여 accumulator에 prev신호로 저장된 후 다음 블록들의 DC 값은 gen_abit 블록 회로를 통하여 입력되는 수와 누진적으로 더하여진다.

gen_abit 블록에서는 DC부가 비트의 그룹 정보로부터 실제적인 부가 비트의 크기를 계산한다. 신호 ssss는 4비트의 그룹 번호를 나타내고, 이로부터 부가 비트, abits의 유효한 MSB를 먼저 찾은 후 MSB가 '1'이면 DIFF가 양수를 나타내므로 abits는 그 크기를 그대로 12비트로 확장하여 출력한다. MSB가 '0'인 경우는 음수이고 이 경우는 DIFF-1에 대한 그룹 위치로부터 크기를 찾아야 한다. 음수 DIFF에 대한 부가 비트는 다음 순서로 구한다.

(1) DIFF-1의 값을 가지고 부가 비트로 입력되는 abits에 +1을 하면, DIFF 크기의 유효 부가 비트 자리 수에 해당하는 abits만이 2의 보수로 표현된다.

(2) gbits의 MSB는 음수 부호인 '1'로 set시킨다.

(3) gbit[MSB-1]에서 부가 비트의 바로 윗자리까지 음수에 대한 2의 보수를 위하여 모든 비트를 '1'로 set한다.

앞절에서 사용된 DC값에 대하여 이 과정을 행하면 음수인 -13에 대한 DIFF-1은 11비트로 00000000010으로 입력될 것이고, 이중 유효 비트는 밑줄 친 0010에 해당한다. 0010의 MSB가 0이므로 이 부가 비트는 음수이고 0010에 1을 더하여 0011을 구한다. 8비트의 나머지 ggits는 1로 채워져서 12비트, 111111110011로 덧셈기에 입력된다.

이때 accumulator는 이전 블록의 DC값인 105인 000001101010를 유지하고 있으므로 2의 보수 덧셈에서 캐리는 무시하여 합의 결과인 000001011100을 내므로 이 블록의 dc_val는 92를 복원하게 된다.

부가 비트 정보를 생성하는 gen_abit 블록은 AC

계수에 대하여서도 동일하게 동작한다. AC 계수의 그룹 크기나 부가 비트의 수는 DC와 다르지만 입력 비트 수는 DC와 동일하므로 같은 블록을 멀티플렉싱하여 사용할 수 있다.

RLC 모듈은 필요한 수만큼의 zero를 생성하고 nonzero 계수인 DC 또는 AC 계수와 함께 zigzag scan의 순서로 출력한다. Huffman decoder로부터 해독된 RS 정보에는 ZRL, EOB 또는 RS의 값에 따라 필요한 0의 개수가 포함되어 있다. 또한 codec의 단위는 8x8의 64화소 단위인 한 블록으로 수행이 되므로 한 개의 블록의 크기를 추적하고 있어야 한다. 이로부터 EOB 신호를 인식하게 되면 현재 화소에서 나머지 모든 화소는 0으로 채워지게 하고, 그 후의 화소는 다음 블록의 DC 계수로 인식되어지게 한다. (그림 5)는 이러한 과정을 수행하는 RLC 모듈에 대한 구조도이다.

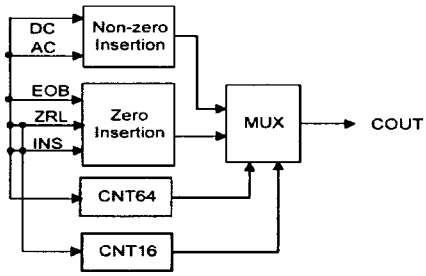

그림 5. RLC 모듈 구조도

Zero insertion module ZRL 신호인 경우 16개의 0을, INS 신호인 경우는 그때의 R값에 해당하는 0을 삽입하게 된다. 또한 EOB 신호인 경우, 블록의 나머지 모든 계수는 0으로 채워서 출력한다.

CNT64는 mod-64 카운터로서 DC계수부터 모든 계수를 출력할 때마다 1씩 증가하여 한 개 블록의 끝에서 carry 신호를 발생한다.

CNT16은 mod-16 카운터로서, ZRL이나 INS 신호에 대하여 삽입할 0의 수를 load한 후 down 카운터로 동작한다.

3.3 Inverse Zigzag Scan 모듈 설계

Huffman Decoder와 RLC decoder를 거쳐 생성되는 복원된 데이터는 zigzag scan 순서로 나타나게 된다. 이를 역양자화기에 입력하기 전에 다시 raster scan 순서로 변환하는 단계가 Inverse zigzag scan module이다.

RLC Decoder의 출력은 64개의 한 블록 단위로 출력되고, 역양자화를 위하여서는 raster scan 순서의 데이터가 필요하게 되므로 한 개의 블록이 완전히 복화화되기 전에는 역양자화의 단계는 시작할 수 없다. 따라서 그림 6과 같이 두 개의 64 워드 RAM 블록을 bank 단위로 하여 복호화된 zigzag scan 순서의 계수를 먼저 저장하게 된다. 한 개의 블록에 대한 계수 저장이 완료된 후, 이 블록으로부터 zigzag scan 주소에 대응하는 raster scan 주소로 데이터가 역양자화기로 입력되게 된다. 한편, 두 번째 블록에 대한 zigzag 데이터는 bank0가 양자화기에 읽혀지고 있는 동안 bank1에 저장되어 진다. Bank0 데이터의 출력이 완료되는 시점에서 다시 bank1로 스위칭되어 출력이 시작된다.

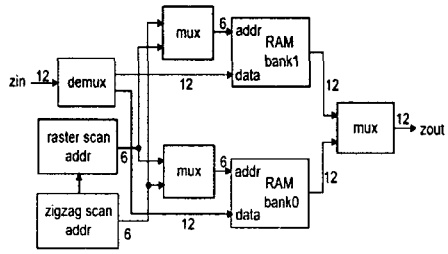

그림 6. Inverse Zigzag Scan 모듈 구조도

그림 6에서 zigzag 계수 입력, zin은 디멀티플렉서를 통하여 bank0과 bank1의 RAM data port로 연결된다. 데이터 분배의 선택은 mod-64 카운터의 carry flag로서 제어한다. Bank0과 bank1의 주소선은 zigzag 주소와 raster 주소가 각각 mod-64 마다 스위칭된다.

표 1. Inverse zigzag scan 주소 변환표

zaddr	raddr	zaddr	raddr	zaddr	raddr	zaddr	raddr
00	00	10	0C	20	23	30	3A
01	01	11	13	21	2A	31	3B
02	08	12	1A	22	31	32	34
03	10	13	21	23	38	33	2D
04	09	14	28	24	39	34	26
05	02	15	30	25	32	35	1F
06	03	16	29	26	2B	36	27
07	0A	17	22	27	24	37	2E
08	11	18	1B	28	1D	38	35
09	18	19	14	29	16	39	3C
0A	20	1A	0D	2A	0F	3A	3D
0B	19	1B	06	2B	17	3B	36
0C	12	1C	07	2C	1E	3C	2F
0D	0B	1D	0E	2D	25	3D	37
0E	04	1E	15	2E	2C	3E	3E
0F	05	1F	1C	2F	33	3F	3F

Zigzag scan 순서에 대응하여 나타나는 raster scan 변환 모듈은 각 위치에 따른 주소를 일대일로 대응하여 생성하면 된다. (표 1)에 zigzag 순서대로

생성되어지는 주소에 대하여 변환되어질 raster scan의 주소값을 16진수로 나타내었다. 이러한 변환표에 의하여 raster scan 주소 생성 회로는 zigzag scan 주소 6비트를 입력으로 하는 단순한 조합회로로 설계되어질 수 있다.

3.4 IDCT 설계

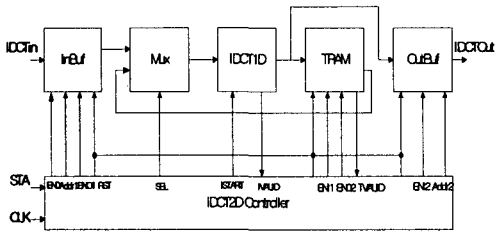

그림 7. ICDT 모듈 구조도

픽셀당 1clock 사용하여 입력에 대한 최초 데이터를 읽기 위해 8clock이 필요하다. multiplication에 대해 1클록이 사용되고 adder에 대해 2clock이 사용된다. 입력에서 데이터가 출력되기까지 총 12(데이터 처리시간)+64(8x8블록)=76clock이 소요된다[8].

4. JPEG Decoder 구현

4.1 System Block Diagram

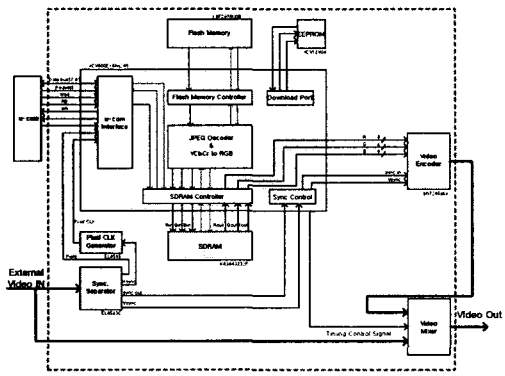

그림 8. 시스템 구조도

그림 8에서는 제안한 JPEG 이미지 출력 보드를 실현하기 위한 하드웨어 구조를 나타내었다[9].

본 논문에서 구현한 보드에 적용된 디바이스(Device)는 표 2와 같다.

표 2. 적용된 디바이스

Part	Part Number	Spec.	Remarks
SDRAM	KS643232F	64M	Samsung
Sync Separator	EL4583C	Sync, burst, field out	ELANTEC
Pixel CLK Gen.	EL4585	27MHz out	ELANTEC
Sub Carrier Freq. Gen.	Oscillator	3.579545MHz	
DAC	ADV7120	Video RGB 24bit input	Analog Device
Video Encoder	bh7240akv	Digital RGB input	Rohm
FPGA	XCV600EHQ 240	60k gate	Xilinx
EEPROM	XCV18V04	4Mbyte	Xilinx
Flash Memory	K9F2808U0B	128M x 8bit	Samsung

그림 9는 위에서 설명한 내용을 실제 보드로 구현한 사진으로 제안한 JPEG 디코더에 대해 실험 및 검증하였다.

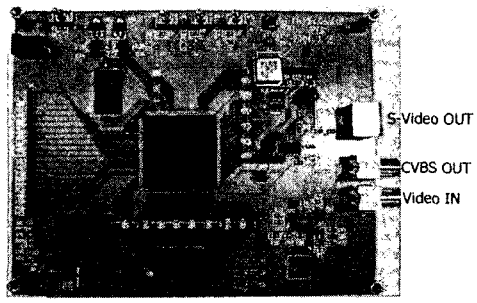

그림 9. 구현한 JPEG 이미지 출력 보드

5. 실험 및 성능 평가

실험은 LG 30인치 LCD TV에 설계한 보드를 연결하여 수행하여다.

그림 10. 실험 환경

그림 11은 본 논문에서 구현한 JPEG 이미지 출력 보드를 통해 디코딩된 JPEG 이미지와 원본 JPEG 이미지를 나타낸 것이다.

(a) 디코딩된 이미지 (b) 원본 이미지

그림 11. 결과 비교

설계된 이미지 출력 보드는 자체 sync generate 기능과 FPGA 내부 RAM을 이용한 character super impose 기능이 있어 다양한 멀티미디어 분야에 응용 가능하다.

참 고 문 헌

[1] J. Rissanen and G.G Langdon, "Aritmetic Coding," IBM J. Res. Develop, vol.23, no.2, pp.149-162, March 1979.

[2] J. Ziv and A. Lempel, *"A Universal Algorithm for Sequential Data Compression,"* IEEE Trans, inform ation Theory, vol. IT-23, pp.337-343, 1977.

[3] ISO/IEC 10918-1, CCITT Recommendation T.81, 1993.

[4] W.K. Pratt, Digital Image Processing, John Wiley & Sons, 1978.

[5] D.A. Huffman, *"A Method for the Construction of Minimum Redundancy Codes,"* Proc. of IRE, vol.40. pp.1098-11-1, Sept. 1952.

[6] Ming T. Sun, *"Design of High-Throughput Entropy Codec"*, VLSI Implementation for Image Communication, edited by P.Pirsch, 1993, Elsevier Science Publishers.

[7] S.M. Lei and M.T. Sun, *" An Entropy Coding system for Digital HDTV Applications,"* IEEE Trans, Circuits and Systems for Video Technology, vol. 1, pp. 147-155, March 1991.

[8] Jechang Jeong and Jae Moon Jo, *"Adaptive Huffman coding of 2-D DCT coefficients for Image sequence compression," Image communication*, vol. 7, Elsevier, pp.1-11, Nov. 1995.

[9] Douglas J. Smith, *HDL Chip Design - A Practical Guide for Designing, Synthesizing and Simulating ASICs and FPGAs using VHDL or Verilog,* Doone Publicatins. 1996.

권 병 헌

1987년 한국항공대학교 항공전자공학과 공학사
1989년 한국항공대학교 대학원 항공전자공학과 공학석사
1995년 한국항공대학교 대학원 전자공학과 공학박사
1997년 (주)LG전자 선임연구원
1997 ~ 현재 유학대학 정보통신학과 부교수
관심분야 : 영상신호처리, 통신, 3D displays.

서 범 석

1999년 한국항공대학교 항공전자공학과 공학사
2001년 한국항공대학교 대학원 항공전자공학과 공학석사
2003년 한국항공대학교 대학원 항공전자공학과 박사과정 수료
2003년 ~ 현재 : (주)플렛디스 책임연구원
관심분야 : 영상신호처리, HDL을 이용한 하드웨어 디자인