SystemC를 이용한 JPEG 인코더 / 디코더의 구현

유명근, 송기용
충북대학교 컴퓨터공학과

Implementation of JPEG Encoder and Decoder with SystemC

Myoung-Keun You, Gi-Yong Song
Dept. of Computer Engineering, Chungbuk National University

요 약

본 논문에서는 SystemC를 이용하여 데이터를 압축하는 JPEG의 인코더와 디코더 구현에 대하여 기술한다. SystemC는 SoC의 설계생산성을 높이기 위해 high-level abstraction에 기반하여 시스템을 모델링하고 명시하는 시스템 수준 설계 언어이고, JPEG은 DCT와 Huffman 코드를 이용하여 정지영상 정보를 압축하는 알고리즘이다. 설계된 JPEG 인코더와 디코더 모듈의 동작을 검증하기 위하여 인코더 모듈에 16×16 크기의 픽셀 RGB 데이터를 입력하고, 디코더 모듈에 인코더 모듈의 출력을 입력으로 연결하여 최종 출력되는 데이터를 비교 및 분석하여 확인하였다.

I. 서론

최근 IT 분야의 흐름은 통신, 방송, 컴퓨터, 가전 등 전자 산업의 모든 기술이 하나로 융합되는 형태이며, 멀티미디어와 관련하여 과거의 서비스 보다는 향상된 서비스가 요구되고 있다. JPEG(Joint Photographic Experts Group) [1-3]이라는 용어는 그래픽 이미지 압축에 관한 표준을 의미하며, 현재 가장 많이 쓰이는 정지화면 영상의 규격 중 하나이다. JPEG에 규정되어 있는 압축 방법은 크게 DCT(discrete cosine transform) 압축 방법, 점진적 전송이 가능한 압축 방법, 계층 구조적 압축 방법, 그리고 무손실 압축 방법으로 나누어진다. 그러나 일반적으로 JPEG으로 영상을 압축하여 저장한다고 하면 DCT를 기반으로 한 기본 압축 방법으로 압축하여 저장하는 것을 의미한다. JPEG 파일은 자연 영상을 손실시켜 높은 압축률로 압축하면서도 눈으로는 그 차이를 알 수 없게 하는 방법이기 때문에 네트워크 상에서 많이 사용하는 그림 저장 포맷이다.

시스템 설계 측면에서 설계자의 생산성이 반도체 칩의 복잡도를 제대로 따라잡지 못함으로써 야기되는 설계 생산성과 칩 복잡도의 격차는 점점 넓어지고 있는 동시에 time-to-market은 점점 짧아지고 있는 추세이다. 이에 단일칩 시스템 설계는 효과적인 해결책으로 주목되고 있으며, 이러한 현실에 부응하여 SoC의 설계생산성을 높이기 위해 현재 high-level abstraction에 기반하여 시스템을 모델링하고 명시하는 SystemC [4-8]와 같은 시스템 수준 설계 언어가 대두되고 있다. SystemC를 이용한 설계는 SystemC 모델링(modeling), 시뮬레이션(simulation), 점진적 설계 구체화(progressive refinement), 합성(synthesis)의 전체설계를 하나의 프로세스로 진행하게 된다.

본 논문에서는 SystemC를 이용하여 DCT 변환, 양자화, Huffman 인코딩을 수행하는 JPEG 인코더와 DCT 역변환, 역양자화, Huffman 디코딩을 수행하는 JPEG 디코더 설계에 대하여 기술한다. 설계된 JPEG 인코더와 디코더 모듈의 동작을 16×16 크기의 필셀 RGB 데이터를 입력으로 주었을 때 출력 데이터를 비교 및 분석하여 검증하였다.

II. Preliminary

1. SystemC

SystemC [4-8]는 하드웨어 모델링용 라이브러리를 포함시킨 C++ 클래스 라이브러리와 시뮬레이션 커널로 구성된 시스템 수준 설계 언어로 high-level abstraction을 이용한 하드웨어 소자, 소프트웨어 모듈, 그리고 이 둘이 결합된 시스템을 co-design 할 수 있는 설계 환경을 제공한다. SystemC는 병렬성을 프로세스로 모델링하고, 시스템의 구조는 계층구조를 지원하는 모듈(module)과 포트(port), 그리고 채널(channel)등으로 모델링하며, 시간 개념의 클럭을 지원한다.

시스템 설계 언어인 SystemC를 이용한 설계는 그림 1에 보이는 바와 같이 SystemC modeling, simulation, refinement, synthesis의 전체설계를 하나의 프로세스로 진행하게 된다. SystemC는 시스템 수준에서 모델을 설계

하기 시작해서 점진적인 설계 구체화를 통해 RTL 수준까지 단일 언어 환경에서 설계할 수 있도록 해준다. 시스템 설계의 최종 단계는 결국 단위 모듈의 통합이라고 볼 때, 시스템 모델링과 사양 설정의 초기 단계에서부터 구현까지 점진적인 추상화 수준 변환이 가능한 단일 언어의 설계 환경을 갖는다는 것은 SystemC의 큰 장점이라 할 수 있다.

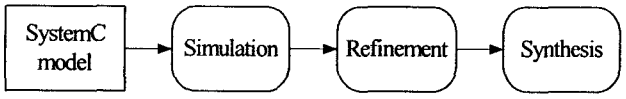

그림 1. SystemC 설계 방법

시스템은 다양한 추상화 레벨에서 모델로 만들어 질 수 있다. 시스템 수준 설계에서 계산 모델 중의 하나인 TLM(transaction-level modeling)은 functional unit 또는 communication mechanism에 대한 구현과 모듈간 통신을 별개로 다루는 high-level modeling 방식이다. TLM에서는 데이터 통신의 실제 구현보다는 그 기능성이 더욱 중요시 되는 특성으로 인하여, pin-accurate-level modeling과 같은 상세한 입출력 기반 인터페이스 방식에 비하여 불필요한 수행을 제거함으로써 빠른 시뮬레이션 속도를 얻을 수 있다는 장점을 얻을 수 있다. 이러한 TLM의 장점은 시스템 설계의 초기 단계부터 시스템 구성이 가능하도록 하여 다양한 구현 방법들의 장단점을 시험해보고 검토할 기회를 제공하며, 또한 구체적인 하드웨어 모델이나 구현된 설계물이 마련되기 이전부터 소프트웨어의 유효성을 충분히 빠른 속도와 정확성을 갖고 검증할 수 있는 기회를 제공한다. 시스템 모델의 추상화 수준을 그림으로 나타내면 그림 2와 같다.

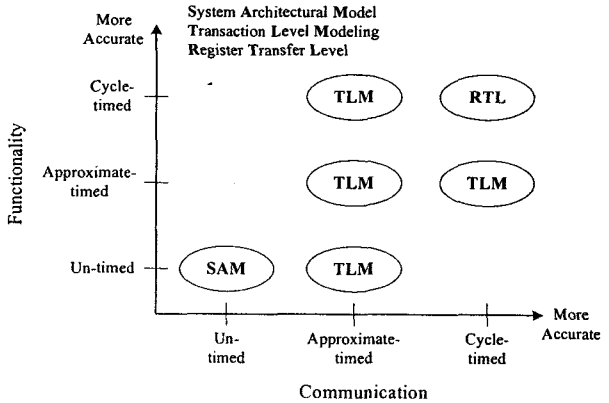

그림 2. 시스템 모델의 추상화 수준

2. JPEG

JPEG 압축 과정 [1-3]은 아래와 같으며 복원 과정은 압축 과정의 역으로 진행되며, 이를 도식화 하면 그림 3과 같다.

- 영상의 컬러 모델을 YCbCr 모델로 변환
- 2×2 영상 블록에 대해 평균값을 취해 색차(chrominance) 신호 성분을 다운 샘플링
- 각 컬러 성분의 영상을 8×8 크기의 블록으로 나누고, 각 블록에 대해 DCT 변환을 수행
- 각 블록의 DCT 계수를 시각에 미치는 영향에 따라 가중치를 두어 양자화
- 양자화된 DCT 계수를 Huffman 코딩 방법에 의해 인코딩하여 파일로 저장

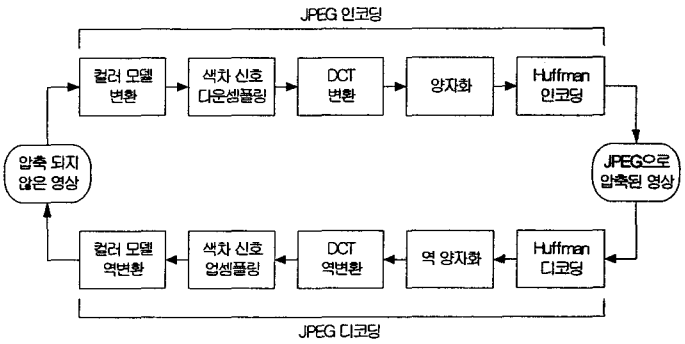

그림 3. JPEG 영상 압축과 복원 과정

1) 컬러 모델 변환

JPEG 압축을 위해서는 일단 컬러 모델을 YCbCr 모델로 변환한다. 그 이유는 YCbCr 모델에서 Y 데이터는 시각적으로 눈에 잘 띄고 Cb와 Cr 데이터는 시각적으로 잘 띄지 않는 정보를 담고 있는 성질이 있어서, Y 데이터는 손실시키지 않고 Cb와 Cr 데이터를 손실시키면 눈으로 봤을 때 화질의 차이를 별로 느끼지 않으면서 정보의 양을 줄일 수 있기 때문이다. RGB 모델과 YCbCr 모델 사이의 변환식은 식 (1)과 같다.

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ Cb &= -0.168R - 0.3313G + 128 \\ Cr &= 0.5R - 0.4187G - 0.0813B + 128 \end{aligned} \quad (1)$$

2) 색차 신호 성분 다운 샘플링

Cb와 Cr 데이터는 시각적으로 눈에 잘 띄지 않는 정보를 담고 있기 때문에, Y 데이터는 모두 저장하고 Cb와 Cr 데이터는 2×2 크기의 블록에서 한 개씩만 저장하고 나머지는 전부 버림으로써 영상 정보의 절반을 줄인다.

3) DCT 변환

DCT 변환의 장점은 변환 전에는 화면에 불규칙하게 퍼져 있던 화소 값이 변환 후에는 저주파 항 쪽으로 집중되는 경향이 있다는 것이다. 따라서 고주파 항들을 버리는 조작을 통해 정보 손실이 거의 없이도 정보 압축을 할 수 있게 된다. 각각의 YCbCr 성분을 8×8 블록으로 나누어 각 블럭에 대해 DCT 변환을 수행하게 된다.

4) 양자화

DCT 변환 후 생성된 DCT 계수에서 고주파 항들을 버리는 위하여 사용되는 방법이 양자화이다. 양자화는 각 주파수의 성분 값을 양자화 스텝 사이즈 값으로 나누어 나머지를 반올림하여 없애는 것이다.

5) Huffman 코딩

양자화된 DCT 계수는 자체로서 압축 효과를 갖지만 압축률을 더 높이기 위해서 Huffman 코딩으로 다시 한 번 압축하여 파일에 저장한다. Huffman 코딩은 무손실 압축 방법의 하나로 이를 이용하여 압축한 후 복원하면 원 데이터가 그대로 복원된다.

III. SystemC를 이용한 JPEG 인코더/디코더 모델링

1. 인코더 모듈

인코더 모듈의 구조를 그림 4에 보인다. 그림의 큰 사각형과 작은 사각형은 각각 8×8 크기의 블럭 데이터와 1-바이트 데이터를 나타낸다.

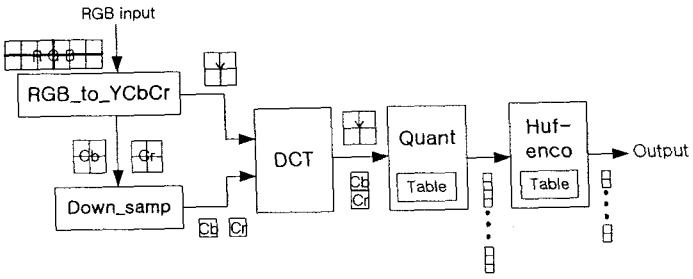

그림 4. 인코더 모듈의 구조

인코더 모듈은 크게 5개의 하위 모듈로 구성되며 각 모듈의 동작은 다음과 같다.

- **RGB_to_YCbCr 모듈** : 16×16 크기의 픽셀 RGB 데이터를 입력 파일로부터 읽어들여 각 픽셀의 RGB 데이터를 식 (1)의 변환식을 이용하여 YCbCr 데이터로 변환한다. Y 데이터는 DCT 모듈로 전송하며, Cb와 Cr 데이터는 Down_samp 모듈로 전송한다.
- **Down_samp 모듈** : 색차 신호를 다운 샘플링하는 모듈로서 입력 받은 8×8 크기의 Cb 블럭 4개와 Cr 블럭 4개를 각각 1개의 Cb 블럭과 Cr 블럭으로 정보를 감소시켜 DCT 모듈로 전송한다. 다운 샘플링의 방법으로 4개의 Cb 블럭과 Cr 블럭에 대하여 2×2 크기의 블럭 평균값을 저장하는 방식을 사용한다.
- **DCT 모듈** : 입력받은 6개의 8×8 크기의 블럭에 대하여 DCT 변환을 수행한다. DCT의 변환식은 식 (2)와 같다. DCT 계수는 DC(direct current) 성분과 AC(alternate current) 성분으로 구분되며, 그림 5에서 b_1이 DC 성분, 나머지 데이터가 AC 성분에 해당된다.

$$X(u,v) = \frac{1}{4}C(u)C(v)\{\sum_{i=0}^{7}\sum_{j=0}^{7}x(i,j)\times \cos(\frac{(2i+1)u\pi}{16})\cos(\frac{(2j+1)v\pi}{16})\} \quad (2)$$

- **Quant 모듈** : 입력받은 모든 8×8 크기의 블럭에 대하여 DCT 계수를 양자화하는 모듈로서 Y 데이터는 휘도 양자화 테이블을 이용하여 양자화하며, Cb와 Cr 데이터는 색차 양자화 테이블을 이용하여 양자화한다. 양자화한 후 그림 5와 같이 데이터를 지그재그 순으로 Huf-enco 모듈로 전송한다.

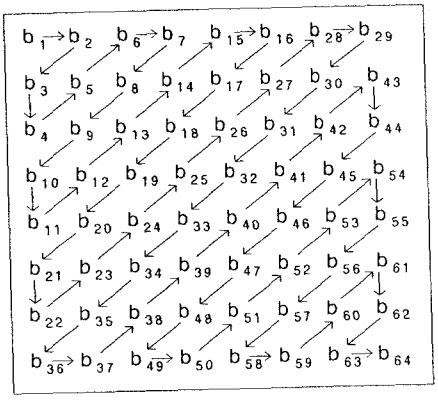

그림 5. 데이터 순서

- **Huf-enco 모듈** : DC 성분과 AC 성분에 대하여 부호화하는 방법이 다르다. 인접한 8×8 블록의 DC 성분 간에는 상관 관계가 크기 때문에 DC 성분은 예측 부호화를 이용하여 부호화한다. Huffman 부호화는 중간 심볼 시퀀스를 생성하고 이를 Huffman 테이블을 이용하여 이진 시퀀스로 변환하는 2단계로 수행된다.

  AC 성분들은 심볼1(runlength, size)과 심볼2(amplitu- de)로 구성되는 중간 심볼 시퀀스를 생성한 후 Huffman AC 코드 테이블을 이용하여 부호화하며, DC 성분들은 실볼1(size)과 심볼2(amplitude)로 구성되는 중간 심볼 시퀀스를 생성한 후 Huffman DC 코드 테이블을 이용하여 부호화한다. 심볼1의 runlength는 0이 아닌 AC 계수 앞에 있는 연속적으로 0인 AC 계수들의 개수이며, size는 계수를 부호화하는데 필요한 비트 수의 카테고리에 해당한다.

2. 디코더 모듈

디코더 모듈의 구조를 그림 6에 보인다. 디코더 모듈은 크게 5개의 하위 모듈로 구성되며 각 모듈의 동작은 다음과 같다.

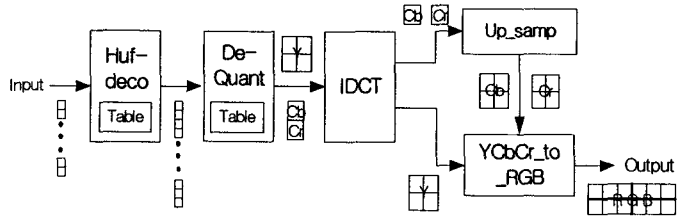

그림 6. 디코더 모듈

- **Huf-deco 모듈** : 압축된 데이터를 입력 받아 Huffman 코드 테이블을 이용하여 이진 시퀀스를 심볼 시퀀스로 변환하고, 심볼들을 DCT 계수로 변환한다. 결과를 De-Quant 모듈로 전송한다.
- **De-Quant 모듈** : Huf-deco 모듈로부터 받은 데이터를 그림 5의 8×8 크기의 블록으로 저장한 후, 역양자화 테이블의 값을 곱한다.
- **IDCT 모듈** : 입력받은 6개의 8×8 크기 블럭에 대하여 IDCT 변환을 수행한다. 변환식은 식 (3)과 같다.

$$x(i,j)=\frac{1}{4}\{\sum_{u=0}^{7}\sum_{v=0}^{7}C(u)C(v)X(u,v)^{*} \cos(\frac{(2i+1)u\pi}{16})\cos(\frac{(2j+1)v\pi}{16})\} \quad (3)$$

- **Up_samp 모듈** : 입력받은 8×8 크기의 Cb 블럭과 Cr 블럭에 대하여, 블럭을 구성하는 각 데이터에 대하여 2×2 크기의 블럭에 복사하는 방법을 통하여 업샘플링을 수행하여 얻어진 4개의 Cb 블럭과 Cr 블럭을 YCbCr_to_RGB 모듈로 전송한다.
- **YCbCr_to_RGB 모듈** : 식 (4)의 변환식을 이용하여 입력받은 Y, Cb, 그리고 Cr 데이터를 RGB 데이터로 변환한 후 출력 파일로 쓴다.

$$\begin{aligned} R &= Y+1.402(Cr-128) \\ G &= Y-0.34414(Cb-128)-0.71414(Cr-128) \\ B &= Y+1.772(Cb-128) \end{aligned} \quad (4)$$

IV. 동작 검증

III절에서 설계한 JPEG 인코더와 디코더 모듈에 16×16 픽셀의 RGB 데이터를 적용하여 최종적으로 출력되는 데이터를 비교 및 분석하여 각 모듈의 동작을 검증하였다.

아래의 두 개의 행렬 O와 D는 각각 DCT 모듈로 입력되는 8×8 크기의 블럭과 IDCT 모듈에서 출력되는 8×8 크기의 블럭을 나타낸다. 설계된 모듈의 동작 검증은 원본 데이터인 O 행렬과 인코드/디코드되어 얻어진 데이터 D 행렬의 비교 및 분석을 통하여 이루어졌으며, Huf_enco 모듈에서 AC 성분 중 2개의 부호화 과정을 그림 7에 보인다.

$$O=\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 66 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 75 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix} \quad D=\begin{bmatrix} 58 & 64 & 67 & 64 & 59 & 62 & 70 & 78 \\ 56 & 55 & 67 & 89 & 98 & 88 & 74 & 69 \\ 60 & 50 & 70 & 119 & 141 & 116 & 80 & 64 \\ 69 & 51 & 71 & 128 & 149 & 115 & 77 & 68 \\ 74 & 53 & 64 & 105 & 115 & 84 & 65 & 72 \\ 76 & 57 & 56 & 74 & 75 & 57 & 57 & 74 \\ 83 & 69 & 59 & 60 & 61 & 61 & 67 & 78 \\ 93 & 81 & 67 & 62 & 69 & 80 & 84 & 84 \end{bmatrix}$$

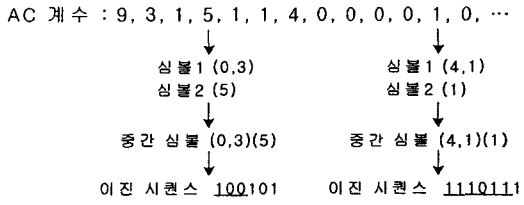

그림 7. AC 성분 부호화 과정

V. 결론

본 논문에서는 DCT와 Huffman 코드를 사용하는 JPEG의 인코더와 디코더를 SystemC를 사용하여 행위 수준(behavioral-level)으로 모델링 하였다. 설계된 인코더 모듈과 디코더 모듈은 각각 5개의 하위 모듈로 구성되며 각 모듈 사이의 상호 통신은 sc_buffer 채널을 통하여 이루어진다.

향후 연구과제로는 JPEG 포맷의 마커 코드들을 이용하여 압축 및 복원하는 인코더/디코더 설계와 동영상 압축 방법인 MPEG(Moving Picture Experts Group)의 인코더/디코더 설계가 있다.

참 고 문 헌

[1] Rafael C. Gonzalez, Richard E. Woods, *Digital Image Processing*, Addison-Wesley, 1993.

[2] Greory A. Baxes, *Digital Image Processing:Principles and Applications*, John Wiley & Sons., Inc., 1994

[3] 정제창 역, *그림으로 보는 최신 MPEG*, 교보문고, 1997.

[4] J. Bhasker, *A SystemC Primer*, Star Galaxy Publishing, 2002.

[5] David C. Black, Jack Donovan, *SystemC:From the Ground Up*, Eklectic Ally, Inc., 2004.

[6] Thorsten Grötker, Stan Liao, Grant Martin, Stuart Swan, *System Design with SystemC*, Kluwer Academic Publishers, 2002.

[7] OSCI, *SystemC version 2.0 User Guide*, Synopsys, Inc., 2001.

[8] 기안도, *SystemC 시스템모델링 언어*, 대영사, 2004.