고성능 DSP 에서의 H.263 인코더 최적화

*문종려, **최수철, *정선태
*숭실대학교 정보통신전자공학부
**광주대학교 컴퓨터전자통신공학부
e-mail : *palmsjoy@syscon.ssu.ac.kr*

Optimization of H.263 Encoder on a High Performance DSP

*Jong-Ryeo Moon, **Soochul Choi, *Sun-Tae Chung
*Dept. of Electronic Eng., Soongsil University
**Dept. of Electronic Eng., Kwangju University

Abstract

Computing environments of Embedded Systems are different from those of desktop computers so that they have resource constraints such as CPU processing, memory capacity, power, and etc.. Thus, when a desktop S/W is ported into embedded systems, optimization should be seriously considered.

In this paper, we investigate several S/W optimization techniques to be considered for porting H.263 encoder into a high performance DSP, TMS320C6711. Through experiments, it is found that optimization techniques employed can make a big performance improvement.

I. 서론

최근 들어, 디지털 카메라, 임베디드 웹카메라 서버, PDA 등 임베디드 환경에서 영상 압축을 필요로 하는 응용이 많이 생겨 나고 있다. 데스크탑 환경과 달리, 임베디드 환경은 메모리, CPU 프로세싱 능력, 전력 등에 제한을 갖게 된다[1]. 따라서, 이러한 자원 제약이 없는 데스크탑 환경에서 구현된 영상 코덱 등의 소프트웨어를 임베디드 시스템에 이식할 때는 성능, 메모리 사용 및 전력 사용이 최적화되도록, 경우에 따라서는 재설계 및 재구현이 되어야 한다[2]. 임베디드 시스템에서의 소프트웨어 최적화 작업은 컴파일러의 최적화 기법, 시스템의 하드웨어 구조를 고려한 S/W 최적화 기법, 어셈블리 코드 최적화 기법 등이 요구된다[3].

임베디드 시스템에서 영상 코덱 처리는, 디지털 연산처리 계산량 부하가 많으므로 이에 전용적인 DSP 칩을 많이 이용하게 된다[4]. 본 연구원들의 연구실에서 수행하고 있는 '임베디드 웹카메라 서버' 개발에서는 필요한 동영상 압축을 위해 동영상 코덱은 H.263을 사용하고 있으며, 이 압축 코덱의 처리를 위해 고성능 DSP 칩인 Texas Instrument 사의 TMS320C6711를 채택하고 있다[5].

본 논문은 고성능 DSP 칩인 TMS320C6711에 이식하는 H.263 인코더의 성능 최적화에 대한 연구 결과를 기술한다. 본 논문에서 보고되는 결과는 고성능 DSP에 이식되는 S/W 의 최적화 방안에 도움을 줄 것으로 기대된다.

본 논문의 구성은 다음과 같다. 제2절에서는 본 논문 내용의 배경인 H.263 과 TMS320C6711 에 대해 간단히 소개되며, 제3절에서는 고성능 DSP에서의 최적화 방안에 대해 기술한다. 제4절에서는 실험 및 검토 결과가 설명되며, 마지막으로 제5절에 결론이 주어진다.

II. 배경

2.1 H.263

H.263 은 유선전화망(PSTN)이나 셀룰러 전화망과 같이 대역폭이 작은 망에서의 전송을 위한 저비트율 통신용 동영상 압축 코덱 규약으로 ITU 에서 1998년에 표준화되었다[6]. H.263 은 블록 기반 DCT 변환 이용 동영상 압축코덱이며, 압축된 영상 스트림은 Picture, Group of Block(GOB), macroblock(MB), block 의 4개의 계층 구조를 갖는다. Block 은 휘도(luminance; Y), 색상(chrominance; Cb, Cr) 등의 종류가 있으며, 모두 8 픽셀 × 8 픽셀 로 구성된다. Macroblock 는 4개의 휘도 블록(Y1, Y2, Y3, Y4), 2개의 색상 블록 (Cb, Cr) 등 총 6개의 블록으로 구성된다. GOB 는 11개의 MB 로 구성되며, Picture 는 QCIF 의 경우 9개의 GOB 로, CIF 의 경우 36개의 GOB 로 구성된다.

H.263의 영상 압축 모드에는 인트라 모드와 인터모드

가 있다. 인트라모드는 정지화상 압축 JPEG 과 같이 현재 프레임에서의 공간 정보만을 이용하여 압축한다. 이 때, 블록은 DCT를 이용하여 변환한 후, 양자화(Quantization)하고 이후 VLC(Variable Length Code) 코딩을 한다. 인터모드는 현재의 압축 대상 영상 프레임을 이전 영상 프레임과 매크로블록별로 움직임 보상을 한 후 비교하여 그 차이만을 블록별로 DCT 변환 및 양자화를 수행하고 VLC 코딩을 하여 압축한다. 움직임 보상은 움직임 예측(Motion Estimation ; ME)이 요구되며, 움직임 예측에는 현재의 매크로블록과 가장 유사한 매크로블록을 이전 영상 프레임에서 찾는 검색을 필요로 하는 데, 이 검색에 많은 연산량이 요구된다. 또한, 비교되는 이전 영상 프레임은 실제 이전 영상 프레임이 아니고, 코딩된 이전 영상 프레임을 다시 복원하여 만든 영상 프레임이기 때문에, 역양자화(IQ), 역DCT변환(IDCT) 등을 통한 영상 프레임 복원이 필요하며, 이 또한 연산에 많은 시간이 소요된다. 인터모드는 인트라 모드에 비해 일시적으로 보관하여야 할 데이터가 많으므로 메모리 수요가 더 많으며, 복잡한 연산 수행이 요구된다.

H.263 인코더 사양에는 기본 사양과 옵션 사양이 있다. 기본 사양은 baseline 이라 하고, 옵션 사항은 annex 형태로 규약되어 있다. 본 논문에서는 baseline H.263 인코더만을 고려하였다. H.263 등에 대한 보다 자세한 내용은 [5]를 참조하라. 본문에서 사용한 H.263 인코더는 TMN 20 코드로써, 데스크탑용으로 구현되어 있다.

2.2 TMS320C6711

TMS320C6711은 Texas Instruments 사의 TMS 320C6000 계열의 칩으로, floating point를 지원하며, 한 사이클에 최대 8개의 명령을 병렬처리할 수 있는 VLIW(Very Long Instruction Word) 구조를 지원한다. TMS320C6000 패밀리는 2개의 병렬 32 비트 데이터 패스, 16개의 32 비트 레지스터를 가지며, 각 데이터 패스는 4개의 32 비트 기능 유니트(가산기, 16×16 승산기, shifter, load/store unit)를 갖는다. 256 비트 명령이 지원되며, 이는 동시에 수행될 수 있는 8개의 32비트 명령으로 구성된다. C6000 패밀리용 컴파일러는 C코드의 병행처리를 위한 컴파일링을 지원한다. 본 논문에서 사용한 6711은 150MHz CPU 클록 속도를 지원하는 TMS320C6711-150 이다. 6711에서 캐쉬는 L1, L2 의 2계층 구조를 가지며, L1 및 L2 캐쉬 모두 CPU 내부에 위치한다. L1 캐쉬는 프로그램용 및 데이터용의 2가지로 구성되며 크기는 각각 4KB이다. L2 캐쉬는 64KB 의 크기를 갖는 데, 이 L2 캐쉬는 메모리의 일부 또는 전부를 내부램(internal RAM)으로 전용이 가능한 구조로 되어 있다. CPU 외부 메모리로는 2GB까지 지원이 가능하다. 외부 메모리는 EMIF (External memory Interface)를 통해 CPU 및 CPU 내부 메모리와 연결된다. 또한 DMA 가 지원되어 memory-to-memory direct copy가 가능하다. TMS320C6000 패밀리와 6711 에 다한 보다 자세한 내용은 [3,7]를 참조하라.

III. 최적화 방안

3.1 임베디드 시스템에서의 S/W 성능 최적화

일반적으로 임베디드 시스템의 S/W 최적화는 먼저 C 로 작성된 코드에 대해 프로파일링을 한 후, 성능이 충분히 나오지 않으면, C 컴파일러가 제공하는 최적화 기법을 사용하여 가능한 한 성능 최적화를 수행한다. 이러한 개선에도 불구하고 성능이 원하는 만큼 충분히 나오지 않으면 다시 프로파일링을 하고 프로파일링시에 시간이 많이 걸린 루틴들에 대해 다시 최적화되도록 재설계·재구현한다 이후에도 성능이 못미치게 되면, 어셈블리로 해당 루틴들을 다시 코딩하여 최적화를 하게 된다. 또한, 이러한 일반적인 과정과 더불어, 임베디드 시스템 H/W 특성을 고려한 최적화 수행 과정이 필요하다.

3.2 TMS320C6711 H/W 특성을 고려한 S/W 성능 최적화

TMS320C6711에서 고려되어야 하는 H/W 특성은 CPU가 VLIW를 지원하며, 캐쉬가 지원되며, 내부 L2 캐쉬 메모리의 일부 또는 전부가 내부 메모리로 용도 변경이 가능하며, DMA 가 지원된다는 점 등이다.

1) VLIW 특성 고려

8개의 명령을 동시에 수행할 수 있기 때문에 가능한 한, C 루틴을 이 병렬처리를 최대로 이용할 수있도록 설계하여야 한다. 물론 컴파일러가 어느정도 최적화를 수행하지만, 코드 자체가 병렬처리가 불가능하도록 설계되어 있으면, 컴파일러도 어쩔 수없다. 병렬처리를 활용하는 S/W 기법에는 다음들이 고려된다[3].

① Loop unrolling

Loop unrolling은 병렬처리가 가능하도록 루프안에서 수행되는 명령의 개수를 늘리는 기법이다. 루프 분기 논리는 오버헤드이므로 분기의 개수를 줄이는 것이 필요하다. 동시에 여러개(최대 8개)의 명령 수행을 지원하는 TMS320C6711에서, 아래 코드 b)는 코드 a) 보다 수행시간이 대략 1/4 로 줄어든다.

a) for (i=0; i<128; i++)

```
{sum += const[i] * input[128 - i]; }

b) for (i=0; i<128; i+=4)
  {sum += const[i] * input[128 - i];
   sum += const[i+1] * input[128 - (i+1)];
   sum += const[i+2] * input[128 - (i+2)];
   sum += const[i+3] * input[128 - (i+3)];}
```

② Software pipelining

Software pipelining은 루프와 기능 유니트들이 효율적으로 스케쥴되도록 하는 소프트웨어 최적화 기법이다. 즉, 루프안의 여러개의 반복이 병렬로 수행되도록 할 수있게 한다. TMS320C6711의 경우, 8개의 기능 유니트들이 동시에 사용될 수있다. C 코드 루프 구조를 잘 변경하면, 컴파일러가 효율적으로 루프를 효율적으로 파이프라이닝할 수있다.

2) 캐쉬 히트율(hit ratio) 개선[2]

C에서 다차원 배열은 행(row) 별로 먼저 저장된다.

다음 코드에서, 예를 들어 다차원 배열 b 의 요소

```
for (j=0; j < N; j++)
  for (i=0; i < N; i++)  b[i][j] =0;
```

b[3][4]를 접근할 때, 이 성분이 캐쉬에 없으면(cash miss) 캐쉬 콘트롤러는 이 요소 b[3][4]를 외부 메모리에서 읽어 캐쉬에 가져 온다. 이 때 캐쉬콘트롤러는 b[3][4] 뿐만 아니라, 근방의 요소들(즉, b[3][5], b[3][6], b[3][...] 등)도 미리 캐쉬에 가져다 놓게 된다. 그러나, 다음 루프에서는 b[3][5]를 사용하는 것이 아니라, b[4][4]를 접근하게 되므로, 크기가 매우 큰 다차원 배열의 경우 b[4][4]의 값이 캐쉬에 없기가 쉬워 다시 메모리에서 이 b[4][4] 값을 읽어 오게 되는 등 캐쉬 미스가 자주 발생하게 된다. 따라서, 상기 코드는 다음과 같이 변경하는 게 성능 최적화에 기여한다.

```
for (i=0; i < N; i++)
  for (j=0; j < N; j++)  b[i][j] =0;
```

3) 내부 메모리 활용

TMS320C6711 과 같은 고성능 DSP 는 CPU 성능에 비해 외부 메모리 접근 속도는 느리다. 따라서, C 루틴이 CPU 내부 메모리에서 수행되는 경우가, 해당 루틴이 외부 메모리에서 수행되는 경우보다 성능이 훨씬 좋게 된다.

4) Direct Memory Access

TMS320C6711 은 DMA를 지원한다. DMA 는 CPU와 무관하게 메모리에서 메모리 이동을 지원하는 메카니즘이다. 따라서, DMA를 사용하면 CPU와 동시에 DMA 가 데이터를 이동할 수 있어, 큰 크기의 데이터 이동에 매우 유용하다.

3.3 TMS320C6711 에서의 H.263 인코더 성능 최적화 방안

전형적인 영상 인코더에서는 가장 계산량이 많은 과정은 움직임추정이다. 이 움직임추정 과정중에서는 SAD(Sum of Difference) 연산이 많은 시간을 소모하는 것으로 알려져 있다. 이밖에, DCT, image interpolation, image reconstruction 등의 연산 등도 계산량이 많은 것으로 잘 알려져 있다. 따라서, 이런 연산 수행 루틴을 최적화하는 것이 필요하다. 최적화 방안에는 컴파일러가 제공하는 최적화 방법에 더하여, 해당 루틴에 대해 S/W unrolling, S/W pipelining, 캐쉬히트개선 등을 고려하여 해당 루틴을 재설계하는 방안이 필요하다. 또한, DCT 연산 등을 최적으로 할 수 있도록 구현된 TMS320C6711 DSP 알고리즘 라이브러리 및 이미지 라이브러리 함수들을 사용하는 것이 바람직하다.

TMS320C6000 패밀리와 같은 임베디드 DSP에서의 영상 인코딩 작업은 외부 메모리에 있는 영상 데이터에 대한 접근과 이 외부 메모리에서의 연산이 상당 부분 차지하게 되는 데, 외부 메모리 접근은 상대적으로 CPU 내부의 연산 처리나 내부 메모리 접근 속도에 비해 느리기 때문에 상기 연산 수행 루틴들(SAD, DCT, image interpolation, image reconstruction 등)을 내부 메모리에서 수행되도록 코드를 재배치하도록 검토하는 게 필요하다. 원하는 충분한 성능이 나오지 않는 경우, 연산 수행에 시간이 많이 걸리는 상기 루틴들을 어셈블리로 코딩하는 방안도 필요하다. 또한, 많은 영상 데이터 이동에는 DMA 기능을 이용하여야 한다.

IV. 실험 및 결과 검토

4.1 실험환경

본 논문에서는 Texas Instruments 의 6711 DSK를 사용하였다. 6711 DSK 는 평가보드 및 호스트에서의 통합 개발환경인 Code Composer Studio를 지원한다. 평가보드에서는 외장 메모리로 16 MB 의 SDRAM을 사용한다. Host 와 평가보드사이의 통신은 JTAG 포트와 패러럴 포트를 지원한다.

사용한 H.263 인코더는 TMN 2.0 으로 소스 코드의 크기는 256KBytes 이다. 실험에서는 baseline 만을 테스트하였고 선택한 영상 포맷은 QCIF 이며, 테스트 영상은 테스트 영상으로 많이 쓰이는 foreman 이다. 또한, 움직임 벡터 추정에는 전탐색(full search) 방법을 선택하였다.

4.2 실험 내용 및 결과 검토

제3절에서 논의한 최적화 방안중 소스코드를 수정하

여야 하는 loop unrolling, software pipelining, cash hit 개선, DMA, 어셈블리 코딩 등에 대한 실험 결과를 포함한 종합적인 최적화에 대한 실험 및 그에 대한 검토 보고는 다음 논문에 자세히 보고하기로 하며, 여기서는 소스코드 변경없이 즉시 테스트가 가능한 최적화 기법에 대한 실험 결과를 보고한다.

실험은 먼저 H.263 basline에 대해 전혀 최적화하지 않은 경우(a), 내부 메모리 이용 최적화를 수행한 경우(b), 컴파일러에서 제공하는 최적화를 수행하여 최적화를 한 경우(c), 내부 메모리 이용 최적화 및 컴파일러 최적화를 모두 수행한 경우(d) 의 4가지에 대해, 주요 루틴 및 인트라모드 인코딩 전체, 인터모드 인코딩 전체에 대해 한번 수행에 소요된 TMS320C6711-150의 CPU 사이클수를 측정하였다(표 1).

	(a)	(b)	(c)	(d)
DCT	17,333	14,231	3,990	3,156
IDCT	122,487	92,245	17,913	15,282
Quantization	62,188	26,883	35,711	20,371
DeQuantization	4,486	3,321	3,153	2,434
Interpolation	2,430,279	2,430,234	1,672,416	1,671,433
SADMB	2,476	1,033	970	745
Motion Estimation	454,182	251,278	139,660	115,253
Intra coding	80,483,453	79,290,167	46,634,218	45,634,864
Inter coding	179,610,967	170,321,787	63,042,919	61,960,781

표 1. TMS320C6711에서의 H.263 최적화

표1에서 보면 최적화에 따라 인터모드 인코딩의 경우, 최적화를 전혀 안한 경우에 비해 수행시간 성능 개선이 189% 증가되었음을 알 수있다. 또한, (d)의 경우에다가 추가적으로 SADMB 루틴만을 어셈블리 코딩하여 테스트해본 실험에서 인터모드 인코딩의 경우 51,325,471 사이클이 소요됨이 관찰되었으며, 이 경우 수행시간 성능 개선이 250% 증가되었음을 알 수있었다. 따라서, 최적화 방법에 따라 인코딩 성능 개선에 큰 차이가 있음을 알 수있다. 추후, 제3절에서 언급한 여러 최적화 기법을 종합적으로 적용하는 경우에 보다 큰 성능 개선이 예측된다.

V. 결론

본 논문에서는 병렬처리를 지원하는 VLIW, 캐쉬 등을 갖는 TMS320C6711과 같은 고성능 DSP 에 H.263 인코더와 같은 영상 코덱을 이식하는 경우에 있어서 고려하여야 할 성능 최적화 방안을 살펴보았다. 실험결과, 최적화 방법에 따라 인코딩 수행 시간 능력의 개선이 많이 이루어 질 수있음을 알 수있었다. 추후 보다 포괄적인 최적화 방법 적용 결과가 보고될 것이다. 본 논문의 결과는 고성능 DSP에 이식되는 S/W의 최적화 방안에 도움을 줄 것으로 기대된다.

참고문헌

[1] P. G. Paulin, et al., "Embedded Software in Real-Time Signal Processing Systems: Application and Architecture Trends," Proc. of IEEE, pp. 419-435, Vol. 85, No.3, Mar. 1997.

[2] P.R. Panda, et al., "Data and Memory Optimization Techniques for Embedded Systems," ACM Transactions on Design Automation of Electronic Systems, pp. 149-206, Vol. 6, No.2, April 2001.

[3] *TMS320C6000 Programmer's Guide.* No. SPRU189E, Texas Instruments, Jan., 2000

[4] H. R. Sheikh, et al., "Optimization of a Basline H.263 Video Encoder on the TMS320C6000," Proc. Texas Instruments DSP Educator's Conference, Aug. 2000.

[5] 이율삼, 백승걸, 정선태, "Design and Implementation of An Embedded Web Camera Streaming Server", 2002년도 대한전자공학회 추계학술대회 논문집, pp. 17-20

[6] G. Cote, et al., "H.263+:Video coding at Low Bit Rates," IEEE trans. Circuits and systems for Video technology, pp. 849-866, Vol.8, No.7, Nov. 1998.

[7] *TMS320C6711 Datasheet,* Texas Instruments,