H.264 정수변환 및 양자화를 위한 VLIW ASIP 프로세서

양승준, 박상현, 허인구, 김용주, 김경원, 백윤흥
서울대학교 공과대학 전기컴퓨터 공학부
e-mail : sjyang, shpark, igheo, yjkim, kwkim@optimizer.snu.ac.kr, ypaek@snu.ac.kr

VLIW ASIP Processor for Integer Transform and Quantization in H.264

Seungjun Yang, Sanghyun Park, Ingoo Heo, Yongjoo Kim, Kyoungwon Kim, Yunheung Paek
School of Electrical Engineering, Seoul National University

요 약

H.264 비디오 코덱은 비디오 어플리케이션 중에서 중요한 역할을 차지하고 있다. 본 논문에서는 H.264 의 여러 과정 중에서 정수 변환 및 양자화 과정을 효과적으로 처리하기 위한 VLIW ASIP 아키텍처를 제안한다.

1. 서론

최근 들어 H.264 코덱과 같은 고성능 미디어 어플리케이션을 임베디드 시스템에서 구동하는 경우가 빈번해지고 있다. H.264 코덱은 상당히 많은 연산을 요구하기 때문에, 임베디드 시스템에서 이를 구동하기 위해서는 이러한 연산을 효과적으로 처리하는 것이 필수적이다.

본 논문에서는 H.264 코덱의 여러 가지 연산 과정 중 정수 변환 및 양자화 연산을 효과적으로 처리하기 위한 VLIW(Very Long Instruction Word) ASIP(Application Specific Instruction set Processor) 프로세서를 제안한다. 본 논문의 남은 부분은 다음과 같이 구성되었다: 2 장에서는 제안된 프로세서의 기본적인 특징을 설명한다. 3 장에서는 효과적인 정수 변환 및 양자화를 위해 추가한 특수 명령어에 관해 설명한다. 4 장에서는 실험 환경 및 결과를 설명한다. 마지막 5 장에서는 전체 내용을 요약하고 결론을 제시한다.

2. 기본 아키텍처

본 논문에서 제안한 프로세서는 4-issue VLIW 아키텍처로써, 각 슬롯마다 16 bit 의 명령어를 읽어 들여 처리한다. 프로세서의 기본적인 특성은 다음과 같다.

- 디지털 신호 처리를 위한 DSP co-프로세서
- 16bit 데이터 및 명령어 path
- 5 단계 파이프라인 (FE, DC, MEM, EX, WB)
- 1 개의 레지스터 파일 (32 개 레지스터)
- 4 개의 내부 메모리
- 정수 변환 및 양자화를 위한 특수 명령어

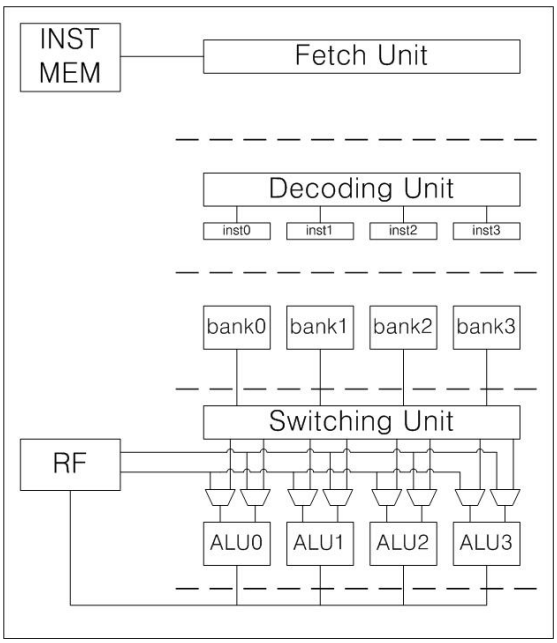

그림 1 기본 아키텍처

각각의 이슈 슬롯은 대부분의 명령어를 실행할 수 있으며, 비교 또는 분기 명령어, 함수 호출과 같은 명령어는 첫 번째 이슈 슬롯에서만 실행될 수 있다.

3. 특수 명령어

본 논문에서 제안하는 프로세서는 4 개의 이슈 슬롯을 가지고 있어, 동시에 최대 4 개의 명령어를 실행할 수 있기 때문에 어플리케이션의 병렬성을 크게 높일 수 있다. 또한 어플리케이션에 최적화 된 명령어를 포함하여 해당 어플리케이션을 보다 더 빠르게 수

행할 수 있다. 이 장에서는 정수 변환 및 양자화를 위해 포함된 특수 명령어를 설명한다.

- ladd, lsub

H.264 정수 변환은 다음과 같은 C 코드를 포함한다.

```
const int s03 = d[i][0] + d[i][3];
const int s12 = d[i][1] + d[i][2];
const int d03 = d[i][0] - d[i][3];
const int d12 = d[i][1] - d[i][2];
```

ladd 및 lsub 명령어는 해당 이슈 슬롯에서 읽어 들인 메모리 값과 다른 이슈 슬롯에서 읽어 들인 메모리 값을 더하거나 빼는 명령어이다. 이 명령어들을 이용하면 위와 같은 C 코드를 한 번에 수행시킬 수 있다.

- shadd, shsub

```
tmp[0][i] =     s03 +     s12;
tmp[1][i] = 2*d03 +     d12;
tmp[2][i] =     s03 -     s12;
tmp[3][i] =     d03 - 2*d12;
```

shadd 및 shsub 명령어는 두 개의 오퍼랜드 중 하나의 값을 지정된 값만큼 시프트 할 수 있다. 이 명령어들을 이용하면 위와 같은 C 코드를 한 번에 수행하는 것이 가능하다.

4. 실험 환경 및 결과

본 논문에서 제안한 프로세서는 Coware 사의 LISA 언어를 이용하여 구현되었다. 이렇게 LISA 언어로 작성한 코드를 Coware 사의 Design Compiler 를 이용하여 Verilog 코드로 변환한 후, Synopsys 사의 합성 툴을 사용하여 합성한 후 게이트 수 및 클록 주파수를 측정하였다. 또한 ModelSim 을 이용하여 정수 변환 및 양자화 연산에 필요한 사이클 수를 계산한 후, 이를 TI 사의 VLIW DSP 인 TMS302C6416 시리즈를 동사의 CodeComposer 를 이용하여 시뮬레이션 한 결과와 비교하였다.

<표 1> 실험 결과

	6416-5e0	6146-6e3	6146-7e3	Our models
Cycle count	168	168	168	80
Clock frequency(Mhz)	500	600	720	300
Throughput (M pixels/s)	31.7	38.1	47.7	41.3

실험 결과, TI 의 TMS320C6416-5e0 에 비교하였을 때는 30%만큼 빠르며 TMS320C6416-7e3 과 비교하였을 때는 13.4%만큼 느리다는 것을 확인할 수 있었다. TMS320C64x 시리즈가 8 개의 이슈 슬롯을 가지고 있다는 것을 생각해볼 때 제안된 프로세서가 보다 더 적은 크기를 가지고 비슷하거나 더 빠른 성능을 보일 수 있었다고 볼 수 있다.

5. 결론

본 논문에서 제안하는 프로세서는 4 개의 이슈 슬롯을 가진 VLIW 프로세서로서, H.264 에 사용되는 정수 변환 및 양자화 연산을 효율적으로 처리하기 위한 특수 명령어들을 포함한다. 실험 결과, TI 의 TMSC3206416 모델과 비교하였을 때 비슷한 성능을 보여주는 것을 확인할 수 있었다.

참고문헌

[1] H. S. Malvar, A. Hallapuro, M. Karaczewicz, and L. Kerofsky, “Low-Complexity Transform and Quantization in H.264/AVC,” IEEE Transactions on Circuits and Systems for video technology, vol. 13, no. 7, July 2003.

[2] Iain Richardson, “H.264/MPEG-4 Part 10 : Transform & Quantization”, H.264 online tutorials, http://www.vcodex.com .

[3] Heng Liao, A. Wolfe, "Available parallelism in video applications," Microarchitecture, IEEE/ACM International Symposium on, pp. 321, 30th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'97), 1997.

[4] TMS320C6000 CPU and Instruction Set Reference Guide, Texas Instruments Inc., Dallas, TX, 2000.

[5] Z. Shen, H. He, Y. Zhang, and Y. Sun, “VS-ISA : A Video Specific Instruction Set Architecture for ASIP design”, Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2006.

[6] Zheng Shen, Hu He, Yanjun Zhang, and Yihe Sun, “A Video Specific Instruction Set Architecture for ASIP design,” VLSI Design, vol. 2007, Article ID 58431, 7 pages, 2007. doi:10.1155/2007/58431

[7] van Eijndhoven, J.T.J.; Sijstermans, F.W.; Vissers, K.A.; Pol, E.J.D.; Tromp, M.I.A.; Struik, P.; Bloks, R.H.J.; van der Wolf, P.; Pimentel, A.D.; Vranken, H.P.E., "TriMedia CPU64 architecture," Computer Design, 1999. (ICCD '99) International Conference on , vol., no., pp.586-592, 1999

Acknowledge

본 연구는 교육과학기술부/한국과학재단 우수연구센터육성사업(R11-2008-007-01001-0), 지식경제부 출연금으로 ETRI, SoC 산업진흥센터에서 수행한 ITSoC 핵심설계인력양성사업, 서울시 산학연 협력사업(10560), 2009 년도 정부(교육과학기술부)의 재원으로 한국과학재단의 국가지정연구실사업(No.2009-0083190)의 지원을 받아 수행되었습니다.