FastImage1300에서의 효율적인 MPEG-4 부호화기 구현

권구현[O] 김성훈 명진수 오승준, 정광수
광운대학교 전자공학부
{lpg9453[O], kungfu, blackart, sjoh}@media.gwu.ac.kr, and kchung@daisy.gwu.ac.kr

Implementing of an efficient MPEG-4 Encoder on FastImage1300

Ku-Hyun Kwon[0] Seong-Hoon Kim Jin-Su Myung Seoung-Jun Oh and Kwangsoo Chung
School of Electronics Engineering, Kwangwoon University

요 약

본 논문은 *FastImage1300*가 제공하는 병렬처리 구조를 사용하여 최적의 MPEG-4 부호화기를 구현하기 위한 프로세서 및 데이터 관리 방법을 제시한다. MPEG-4 부호화기가 가진 병렬성과 순차성을 FastImage1300 특성에 맞추어서 이 시스템이 제공할 수 있는 최대 속도로 동작하는 MPEG-4 부호화기를 구현한다. 이 부호화기를 사용하여 CIF 영상을 MPEG-4 CP@L3로 최대 25fps까지 부호화 할 수 있다.

1. 서 론

최근 디지털 멀티미디어 정보 서비스들이 급속도로 확산되고 있다. 이러한 멀티미디어 정보의 효율적인 전송과 저장을 위해 국제 표준화 기구에서 표준들을 정하고 있으며 그 중의 하나인 MPEG-4 표준은 저가격이며 고성능인 멀티미디어 통신 서비스를 고려하여 유동적으로 기존의 방식 및 새로운 기능들을 모두 지원할 수 있는 부호화 도구와 알고리즘을 제공한다.

그러나 부호화 과정 중에는 움직임 추정 및 DCT(Discrete Cosine Transform)와 같이 많은 계산을 요구하는 부분이 존재하여 단일 프로세서를 사용하여 실시간 구현을 하기가 쉽지 않다. 이러한 문제점을 해결하기 위하여 멀티프로세서를 사용하여 병렬로 처리하는 방법이 제시되었다. 이러한 병렬처리 구조에는 *데이터 분할에 의한 병렬처리*와 *작업 파이프라인에 의한* 병렬처리 방법이 있다.[1]

본 고에서는 FastImage1300이 제공하는 병렬처리 구조를 사용하여 최적의 CP@L3(*Core profile@Level* 3) MPEG-4 부호화기를 구현하기 위한 프로세서 및 데이터 관리 방법을 제시한다. MPEG-4 부호화기가 가진 병렬성과 순차성을 FastImage1300 특성에 맞추어서 이 시스템이 제공할 수 있는 최대 속도로 동작하는 MPEG-4 부호화기를 구현한다.

2. FastImage1300 구조

FastImage1300은 호스트 시스템상에서 4개의 연속적인 비디오 스트림까지 처리할 수 있는 시스템이다. FastImage1300은 그림 1과 같이 4개의 Trimedia TM1300 PCI 미디어 프로세서(TM0~TM3)로 구성되어 있으며 각 프로세서는 8Mbyte에서 16Mbyte의 국부메모리를 갖는다. 또한 FastMem이라는 공유메모리가 PMC 슬롯을 통하여 연결가능하며 이러한 FastMem의 최대 용량은 512Mbyte이다.

3. 병렬처리 구조의 적용

3.1 MPEG-4 부호화 과정의 모듈별 계산량 분석

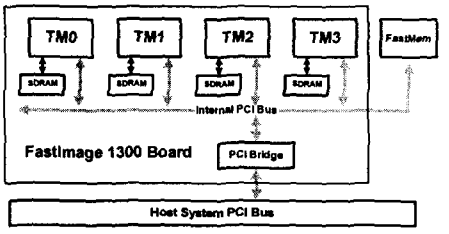

그림 1 FastImage1300 블록도

표 1은 각 모듈별로 최적화 작업을 수행한 후 단일 프로세서를 사용하여 CIF 크기의 Foreman 영상 1프레임을 부호화 하는데 필요한 각 모듈별 최대 수행시간을 측정한 것이다.

표 1 단일 프로세서 상에서의 모듈별 작업 부하 (단위:ms)

Module Name		I VOP	P VOP	B VOP
ME	Integer Pel	N/A	50	29
	Half Pel	N/A	40	17
MC		N/A	13	18
Diff_DCT/Add_DCT		N/A	14	7
Quant/Deguant		6	12	7
IDCT/DCT		6	N/A	N/A
VLC		32	27	16
Total		44	156	94

표 1에서 Diff_DCT는 예측오차를 구하고 DCT를 수행하는 모듈이며, Add_IDCT모듈은 양자화된 예측 오차에 대해 IDCT를 수행하고 참조영상과 합하여 재구성된 영상을 만드는 모듈이다. 표 1에 보였듯이 P VOP의 부호화는 Advance Prediction Mode를 사용하기 때문에 전체과정 중 움직임 추정이 가장 많은 계산량을 요구한다.

3.2 작업 파이프라인에 의한 병렬처리 구조의 적용

MPEG-4 부호화에 대하여 작업 파이프라인에 의한 병렬처리 구조를 적용하는데 있어서 주안점은 작업 부하의 균등한 분할

이다.[2] 이러한 점을 고려하여 FastImage1300상에 MPEG-4의 부호화과정을 구성하면 그림 2와 같이 구성되며 이 때의 각 프로세서에 대한 작업 부하 및 역할은 표 2와 같다.

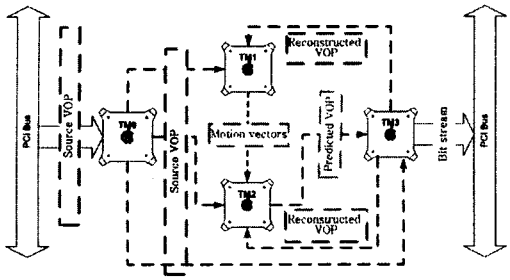

그림 2 작업 파이프라인에 이한 병렬처리 구조

표 2 각 프로세서별 작업 부하 및 역할 (단위:ms)

Processor	P VOP	B VOP	역 할
TM0	35	35	■ Fetch source image.
TM1	50	29	■ Integer Pel ME
TM2	53	35	■ Half Pel ME ■ MC
TM3	53	30	■ Diff_DCT/Add_IDCT ■ Quant/Dequant ■ VLC

3.3 데이터 분할에 의한 병렬처리 구조의 적용

FastImage1300에서 데이터 분할에 의한 병렬처리 구조를 MPEG-4 부호화 과정에 적용하면 그림 3과 같이 구성되며 이 때의 각 프로세서에 대한 작업 부하 및 역할은 표 3과 같다.

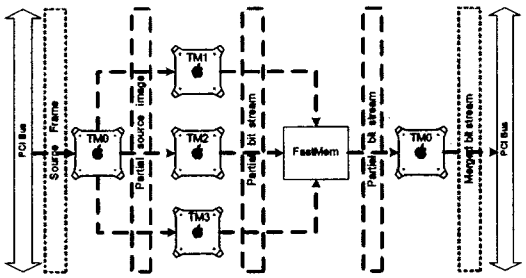

그림 3 데이터 분할에 의한 병렬처리 구조

전체적인 부호화 과정은 마스터 프로세서(TM0)가 원 영상을 읽어온 후 각 프로세서에게 데이터를 분배하며, 분할된 영상은 각 슬레이브 프로세서(TM1/TM2/TM3)에서 각각 부호화 된다.

분할된 영상이 각각의 슬레이브 프로세서에서 부호화될 때 마스터 프로세서는 다음에 부호화될 원 영상을 미리 가져오며, 그 후 각 슬레이브 프로세서가 생성한 비트스트림이 FastMem에 모두 옮겨지게 되면 마스터 프로세서는 각각의 부분적인 비트스트림을 자신의 로컬 메모리로 가져온 후 하나의 비트스트림으로 병합한다. 이 때 다른 슬레이브 프로세서들은 복원된 영상의 교환을 수행하게 된다.

3.4 분 석

표 3 각 프로세서별 작업부하 및 역할 (단위:ms)

Processor	P VOP	B VOP	역 할
TM0	38	38	■ Fetch source image ■ Merge bit stream
TM1	52	31	■ All encoding process ■ exchange reconstruction regions
TM2	52	31	■ All encoding process ■ exchange reconstruction regions
TM3	52	31	■ All encoding process ■ exchange reconstruction regions

병렬처리 구조에서 부호화 알고리즘상의 데이터 종속성이 없다고 가정하면 전체 처리율에 중요한 요소로서 데이터의 I/O (Input/Output)가 있다. 또한 처리되는 대상이 영상이므로 많은 양의 데이터를 고속으로 처리하게 되며 이로 인하여 많은 양의 데이타 I/O가 발생한다. FastImage1300와 같이 여러 개의 프로세서가 동일한 버스를 공유하여 사용하는 구조에서는 데이터의 I/O가 전체 처리율을 제한하는 요소가 되므로 기술한 두 구조에서 발생하는 데이터의 I/O양을 비교해야 할 필요가 있다.

표 4는 CIF 크기의 영상에 대하여 단일 프레임을 부호화할 경우 각각의 병렬처리 구조에서 발생하는 데이터의 I/O양과 부호화 시간을 고려하여 부호화 할 수 있는 최대 프레임율을 나타내었다.

표 4에서 데이터 이동시간에 대한 가정은 FastImage1300에서 버스의 최대 대역폭(70Mbyte/s)을 사용하며 표에 나타내지 않은 사항들은 처리시간이 작아서 무시된다고 가정하였다.

표 4 각 구조별 데이터의 이동량

데이터 종류	작업 파이프라인	데이터 분할
데이터의 이동량 (byte)	918624	190528
데이터 I/O 시간 (ms)	13.1	2.7
부호화 시간 (ms)	53	52
최대 프레임율 (fps)	**15.1**	**18.2**

표 4에서 보인바와 같이 작업 파이프라인에 의한 병렬처리구조는 많은 양의 데이터I/O로 인하여 FastImage1300와 같이 여러개의 프로세서가 동일한 버스를 공유하여 사용하는 구조에는 적합하지 않음을 알 수 있다. 따라서 본 논문에서는 MPEG-4 부호화 과정에 데이터 분할에 의한 병렬처리 구조를 적용하여 구현하였다.

4. 구 현

데이터 분할에 의한 병렬처리 구조를 MPEG-4 부호화 과정에 적용시키기 위해서는 다음 사항을 고려해야 한다.

4.1 고려사항

4.1.1 데이터의 종속성 제거

MPEG-4 부호화 과정 중에는 DC/AC 예측 및 움직임 벡터 예측과 같이 주변 블록을 참조하여 부호화하는 과정이 있다. 하지만 데이터 분할에 의한 병렬처리를 적용하게 되면 동일한 시점에서 분할된 영상이 각 프로세서에서 독립적으로 부호화 되므로 참조할 주변 블록이 다른 프로세서에서 처리될 때까지 대기해야 한다. 이러한 문제를 해결하기 위하여 MPEG-4 Error Resilience Tool중의 Resync Marker를 사용하였다.

4.1.2 재구성된 영역의 교환

MPEG-4 부호화는 기본적으로 UMV(Unrestricted Motion Vector)를 사용하므로 복원영상을 만들 때에는 부득이하게 그림 4와 같은 재구성된 영상의 교환을 수행해야 한다.

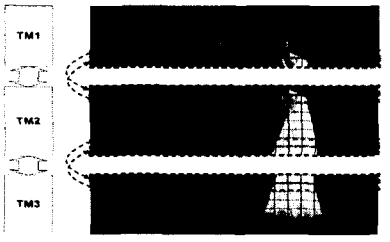

그림 4 영상 복원시 슬레이브 프로세서 사이의 데이터 교환

4.2 시스템 설계

본 논문에서는 위의 분석을 바탕으로 FastImage1300에서 4개의 Trimedia TM1300를 사용하여 MPEG-4 부호화기를 설계하였으며 이에 대한 구조는 그림 3과 동일하며 각 프로세서별 Flow chart는 그림 6에 나타내었다.

5. 실험 결과

본 논문에서 제안한 MPEG-4 CP@L3 부호화기의 성능을 분석하기 위하여 4개의 Trimedia TM1300 프로세서를 사용하여 CIF 크기의 영상을 부호화 하였을 때의 부호화 속도를 표 4에 나타내었으며, Foreman 영상의 부호화시 각 작업별 시간 점유률에 대한 분석을 그림 6에 나타내었다.

표 4 CIF 영상 사이즈에 대한 MPEG-4 부호화 성능(단위:fps)

Test sequence	CBR (384Kbps)	VBR
Akiyo	24	25
Table	21	20
Foreman	20	19

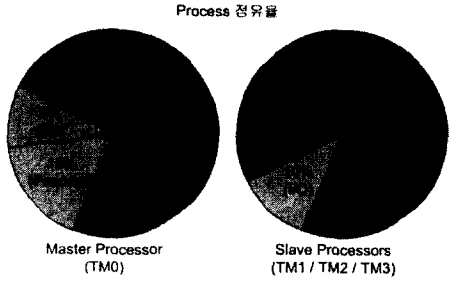

그림 6 각 프로세서별 작업시간 분석

그림 6에서 I/O는 프로세서간 데이터 이동을 의미하며 Process는 프로세서가 실제로 수행한 부호화 작업을 의미한다. 슬레이브 프로세서는 전체 부호화 시간 중 10%를 데이터 I/O에 사용하였으며 11%의 프로세서 유휴시간이 발생하였다. 이러한 유휴시간의 원인은 재구성 영상을 교환하기 위한 각 프로세서의 동기화 및 마스터 프로세서가 영상 데이터를 분배할 때 발생하는 지연에 의해서 발생하였다.

6. 결 론

본 논문은 FastImage1300에서 MPEG-4 CP@L3 부호화 과정의 효율적인 프로세서 및 데이터 관리 방법을 제시하였으며 제시된 방법을 통하여 CIF 영상에 대하여 최고 25fps로 부호화 할 수 있었다.

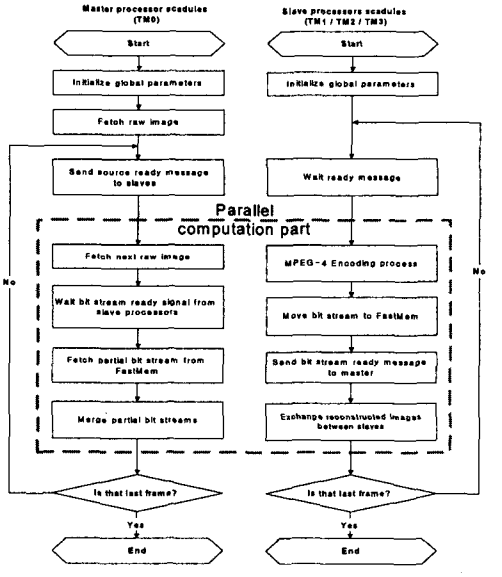

그림 6 마스터 및 슬레이브 프로세서의 Flow chart

감사의 글

본 연구는 한국과학재단 목적기초연구 (R01-2002-000-00179-0) 지원으로 수행되었음

참고문헌

[1]. Downton, A. Crookes, D "Parallel architectures for image processing", IEEE Electronics & Communication Engineering Journal, pp.139-151, Jun 1998.

[2]. Yung, N.H.C; Chu, K.C,. "Fast and parallel video encoding by workload balancing", 1998 IEEE International Conference on System, Man, Cybernetics, pp.4642-4647 Oct 1998.

[3]. Alacron "FastImage 1300 Hardware Users Guide", Sept. 2000.

[4]. ISO/IEC, JTC1/SC29/WG11, N3515, "MPEG-4 Video Verification Model Version 17.0", Jul. 2000.