논문 99-24-6B-23 한국통신학회논문지 '99-6 Vol.24 No.6B

효율적인 프레임 메모리 인터페이스를 통한 MPEG-2 비디오 인코더의 개선

정회원 김 견 수*, 고 종 석*, 서 기 범**, 정 정 화**

An Improvement of MPEG-2 Video Encoder Through Efficient Frame Memory Interface

Kyeounsoo Kim*, Jong-Seog Koh*, Ki-Bum Suh**, Jong-Wha Chong** *Regular Members*

요 약

본 논문에서는 MPEG-2 비디오 인코더를 ASIC 칩으로 구현할 때, 움직임추정기와 함께 대량의 하드웨어 영역을 차지하는 프레임메모리 인터페이스를 개선한 효율적인 하드웨어 구조를 제시한다. 이를 위해 비디오 인코더와 듀얼 뱅크를 가지는 외부 SDRAM 사이의 인터페이스를 효율적으로 처리할 수 있도록 메모리 맵을 구성하고 메모리 액세스 타이밍을 최적화하여 내부 메모리 크기와 인터페이스 로직을 줄였다. 본 설계에는 0.5 m, CMOS, TLM(Triple Layer Metal) 표준 셀 라이브러리가 사용되었으며, 하드웨어 설계 및 검증을 위해서 VHDL 시뮬레이터와 로직 합성 툴이 사용되었고, 기능 검증을 위한 테스트 벡터 생성을 위해서, C 언어로 모델링한 하드웨어 에뮬레이터가 사용되었다. 개선된 프레임 메모리 인터페이스의 구조는 기존의 구조[2-3]에 비해 58% 정도의 면적이 감소했으며, 전체 비디오 인코더에 대해서는 24.3% 정도의 하드웨어 면적이 감소되어, 프레임메모리 인터페이스가 비디오 인코더 전체의 하드웨어 면적에 대단히 심각한 영향을 미친다는 것을 결과로 제시한다.

ABSTRACT

This paper presents an efficient hardware architecture to improve the frame memory interface occupying the largest hardware area together with motion estimator in implementing MPEG-2 video encoder as an ASIC chip. In this architecture, the memory size for internal data buffering and hardware area for frame memory interface control logic are reduced through the efficient memory map organization of the external SDRAM having dual-bank and memory access timing optimization between the video encoder and external SDRAM. In this design, 0.5 m, CMOS, TLM (Triple Layer Metal) standard cells are used as design libraries and VHDL simulator and logic synthesis tools are used for hardware design and verification. The hardware emulator modeled by C-language is exploited for various test vector generation and functional verification. The architecture of the improved frame memory interface occupies about 58% less hardware area than the existing architecture[2-3], and it results in the total hardware area reduction up to 24.3%. Thus, the fact that the frame memory interface influences on the whole area of the video encoder severely is presented as a result.

I. 서 론

멀티미디어 기술의 핵심은 방대한 정보량을 갖는 영상 데이터를 효과적으로 압축, 전송, 저장하는 것이라고 해도 과언이 아니다. 이를 위해 MPEG (moving picture experts group)에서는 움직임추정

* 한국통신 가입자망연구소(kyskim@jungfrau.usc.edu, jkoh@kt.co.kr)
** 한양대학교 대학원 전자공학과(kbsuh@shira.hanyang.ac.kr, jchong@email.hanyang.ac.kr)
논문번호 : 98540-1216, 접수일자 : 1998년 12월
※ 본 연구는 한국통신의 연구비 지원으로 한양대학교와 공동으로 수행되었습니다.

및 보상 기법과 변환 부호화 기법을 기반으로 하는 영상 압축 방식을 표준으로 채택하였다[1]. MPEG-2의 표준화가 완성된 1996년을 전 후하여 실시간 처리를 위한 인코더 및 디코더 칩이 개발되어 활발히 개발되었으며, 최근에는 단일 칩과 전력 소모가 적은 칩을 개발하는데 주력하고 있다. 대부분의 ASIC 칩들은 주변 디바이스, 특히 외부 메모리를 액세스하는데 시간을 많이 소요하고, 제한된 시간 내에 액세스 사이클을 수용해야 하기 때문에, 데이터의 병목현상이 발생한다. 즉, 외부 메모리의 처리 속도가 느리면 이를 보상하기 위해 칩 내부에 데이터 인터페이스를 위한 버퍼가 많이 필요하게 되므로, 처리 속도나 용량에 따라서 내부에서 처리해야 할 인터페이스에 많은 영향을 받는다.

본 논문은 [2-3]에서 개발하였던 비디오 인코더 칩에서 처리속도가 느린 외부 DRAM과의 인터페이스, 내부 버퍼의 비효율적인 사용, 메모리 맵, 그리고 타이밍 스케줄링 등을 개선한 것이다. 이를 위해서 외부에 처리속도가 빠르고 듀얼 뱅크 동작 및 버스트 길이 변화가 가능한 SDRAM을 사용하여 메모리 액세스에 필요한 클럭 수 및 내부 버퍼의 크기를 최소화하였다. 데이터 인터페이스를 위한 버스 폭을 줄여 데이터 버스로 인한 전력 소모를 줄이고, 동시에 내부 메모리 사이의 라우팅 복잡도를 줄여 제어 로직으로 인한 하드웨어 면적을 줄였다.

본 논문의 구성은 다음과 같다. 제 II장에서는 프레임메모리 인터페이스부가 MPEG-2 비디오 인코더에서 차지하는 하드웨어 관점의 비중을 기반으로 효율적인 프레임메모리 인터페이스를 제안하게 된 동기를, III장에서는 SDRAM을 사용한 프레임 메모리 부의 구조개선 방법과 설계 내용에 대해서 설명하며, IV장에서는 설계 결과와 검증 방법을 제시하고, 마지막으로 V장에서는 본 논문의 결론을 내리도록 한다.

II. 비디오 인코더의 구조

본 논문에서 설계 및 구현한 비디오 인코더는 그림 1과 같이, 입력 영상을 받아들이는 입력처리(IP) 모듈, 움직임 보상과 추정을 위한 MEMC(Motion Estimation and Motion compensation) 모듈, DCT와 양자화를 위한 DCTQ 모듈, 가변 길이 부호화를 위한 VLC(Variable Length Coding) 모듈, 부호화 과정을 전체적으로 제어하기 위한 제어(controller) 모듈, 그리고 각 모듈에서 발생하는 데이터를 저장하고 필요한 시기에 전달하는 프레임 메모리 인터페이스 모듈 등 6개의 부분으로 구성되어 있다.

입력 처리 모듈은 외부에서 입력되는 다양한 형식의 영상을 비디오 인코더가 처리하기에 알맞은 형태로 변환해 주는 역할을 한다. 여기서 입력 영상은 4:4:4 형식의 RGB, 4:2:2 형식의 YCbCr, 그리고 ITU-R 601/656 비트 병렬 형식들이며 출력은 4:2:0 형식의 YCbCr이다. 디지털 YC, 아날로그 RGB 신호를 A/D 변환한 4:4:4 형식의 디지털 RGB, ITU-R 601/656 형식의 입력 영상은 4:2:2 형식의 데이터로 변환한 후 프레임 메모리에 잠시 저장된다. 프레임 메모리에 저장하는 것은 4:2:0 형식으로의 변환을 위한 데시메이션 필터링을 위한 것이다. 4:2:0 형식으로 변환된 신호는 MEMC로 입력되며 움직임 추정 및 보상을 수행한 데이터는 비디오 인코더에서 부호화를 위한 입력 데이터로 이용된다. 그리고 입력 영상들은 기준 신호(H_REF, FLD_REF, FRM_REF)를 가지고 있으므로 각 입력 영상에 대한 현재 부호 화할 입력 영상의 기준 신호를 생성하기 위한 기능을 포함한다.

제어 모듈은 비디오 인코더의 각 모듈에게 부호화에 필요한 클럭, 버퍼 상태, 부호화 모드 등을 제어한다. 제어모듈에서 생성된 클럭들은 비디오 인코더 내부의 각 모듈에게 부호화 시작을 알리고, 전송율, 영상 크기 등의 부호화 사양은 호스트 인터페이스로부터 입력받아 인코더 내의 각 모듈에게 전달된다. 그밖에 제어모듈은 인트라/인터, 프레임/필드 DCT, 움직임 추정 모드 및 움직임 추정 범위를 결정하는 역할을 한다.

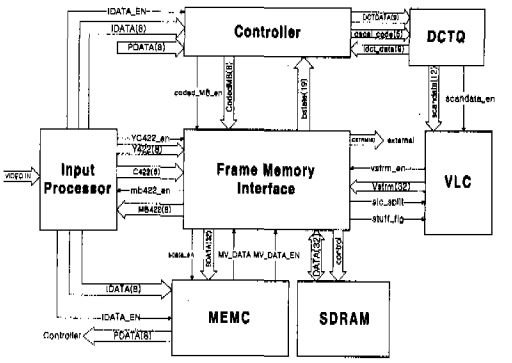


그림 1. 비디오 인코더의 블록도

프레임 메모리 인터페이스 모듈은 정해진 타이밍 스케줄에 따라서 4:2:2 형식의 데이터를 매크로블록 단위로 읽어서 입력처리 모듈로 전달한다. 아울러

제어 모듈을 통해서 부호 화된 데이터를 입력받고, MEMC에 움직임 탐색을 위한 데이터를 제공하며, VLC로부터 부호 화된 비트 열을 입력받아 외부 SDRAM과 인터페이스 역할을 한다.

DCTQ 모듈은 2차원 DCT 및 IDCT, 양자화기 및 역 양자화기, 그리고 주사기로 구성된다. 2차원 DCT 및 IDCT는 행렬과 벡터의 곱셈으로 표시되는 연산을 수행하는 것으로, 하나의 1차원 DCT와 전치 회로로 구성하였다. 양자화는 DCT 결과로 얻어진 계수 값에 양자화 매트릭스 값을 나누어주고, 그 결과를 다시 양자화 스케일 값을 나누어 DCT 계수의 고주파 성분을 제거 시켜 데이터를 압축한다. 나눗셈 연산을 수행하기 위해서 미리 양자화 매트릭스나 양자화 스케일 값의 역수를 미리 구하고 그 결과를 scale-up한 다음에 그 결과와 DCT 계수를 곱하고 다시 그 결과를 scale-down 한다. 역 양자화는 양자화의 역 과정이므로 나눗셈을 포함하지 않아 양자화기 보다 쉽게 구현되었다. 주사기는 제어 모듈에서 입력되는 주사 선택 신호에 따라서 지그재그 및 얼트네이트 주사를 수행한다. 이는 연산 동작이 필요한 것이 아니라 데이터의 순서만 바꾸면 되므로 순차적으로 데이터를 메모리에 저장하였다가 선택된 주사 순서로 데이터를 읽어 내어 출력하면 된다.

VLC는 제어 모듈, DCTQ 모듈로부터 받은 압축 정보 및 압축된 데이터를 고정 길이 및 가변 길이로 부호화하여 MPEG-2의 신텍스에 맞게 부호화하고 프레임 메모리 모듈에 고정 길이로 패킹 된 데이터를 제공한다. 부호화된 비트스트림은 프레임메모리 인터페이스를 통해 외부 SDRAM의 채널 버퍼에 저장되고, 비트스트림은 채널의 요구에 의해 외부로 출력된다.

MEMC 모듈은 휘도 신호에 대한 움직임 추정과 색도신호에 대한 움직임 보상의 역할을 한다. 프레임메모리 인터페이스부로부터 한 개의 매크로 블록당 6 개 참조 매크로 블록을 전송받아 움직임 추정을 수행하며, 움직임 벡터의 계산이 끝나면 해당 매크로블럭에 대한 움직임 보상을 수행하여 예측데이터를 발생시킨다.

여기서 일반 DRAM을 사용한 기존 비디오 인코더의 하드웨어 면적에 대해서 고찰해 보고자 한다. 그림 2는 기존 비디오 인코더의 내부 레이아웃 사진이다. 전체 면적의 약 42%가 프레임 메모리 인터페이스를 위해서 할애되었음을 알 수 있다. 본 논문에서는 이로부터 중요한 2개의 사항에 대해서 언급하고 각각의 해결 방법을 강구하고자 한다. 먼저 기존의 구조에서는 프레임메모리 인터페이스를 위해 내부에 작은 크기의 메모리를 매우 많이 사용하였다. 이것은 인접 메모리와의 라우팅을 복잡하게 하므로 하드웨어 면적을 많이 차지하는 요인이 된다. 다른 하나는 기존 DRAM의 느린 속도로 인하여 내부 메모리의 용량이 증가하고 메모리 스케줄링이 어려워 하드웨어 크기가 증가하였다. 따라서 본 논문에서는 외부에 DRAM 대신에 SDRAM을 사용하여 처리속도를 높였고, 여러 개의 작은 메모리를 사용하지 않고 가능한 한 메모리의 개수를 줄여 인터페이스를 간략화 하였다. 아울러 메모리 맵과 타이밍 시퀀스를 효율적으로 구성하여 하드웨어 크기를 줄였다.

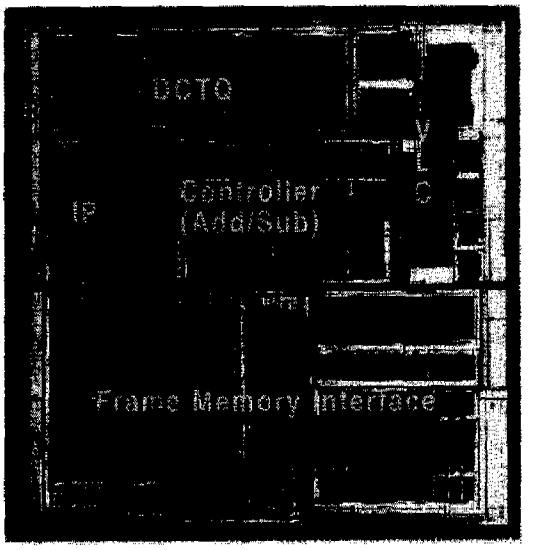


그림 2. 비디오 인코더의 레이아웃 사진

Ⅲ. 프레임메모리 인터페이스

프레임메모리 인터페이스 모듈은 외부 SDRAM을 제어하여 부호화 과정에서 발생하는 데이터를 각 모듈의 요구에 따라 적시에 저장 및 공급하는 역할을 한다. 프레임메모리 인터페이스는 입력되는 데이터를 내부 버퍼에 담아두었다가 SDRAM의 액세스 타이밍 스케줄링에 따라 각 데이터를 32 비트씩 묶어 SDRAM에 저장해두고, 내부 모듈의 데이터 공급 요구에 따라 SDRAM으로부터 읽어와 내부 버퍼에 임시로 저장하였다가 요구하는 모듈에 공급해 준다. 그림 3은 프레임메모리 인터페이스 모듈의 구성을

보인 것으로, 인코더의 각 모듈과 인터페이스를 전담하는 서브 모듈과 서브 모듈의 동작시간을 관리하는 시퀀스 디코더로 구성된다. 각 서브 모듈은 SDRAM과의 데이터를 주고받는 제어기와 내부 버퍼로 구성되며, 프레임메모리 인터페이스의 설계는 내부 버퍼의 크기와 제어기의 로직을 최소화하는 것을 목적으로 하였다. 본 비디오 인코더는 유효 영상 크기가 720x480인 NTSC 비디오 데이터를 RGB 4:4:4, YUV, 혹은 CCIR 601/656등의 형태로 받아들일 수 있다. NTSC 입력 영상의 최대 크기는 858x525이므로 한 매크로 블럭 당 333.6개의 13.5 MHz 클럭을 할당 할 수 있다. 따라서 27 MHz 클럭의 경우 한 매크로블록 기간 동안 667 클럭, 54 MHz의 경우 1334 클럭을 할당할 수 있다.

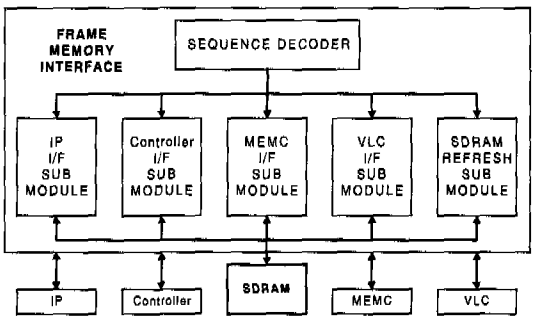
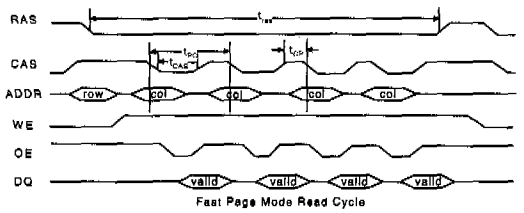
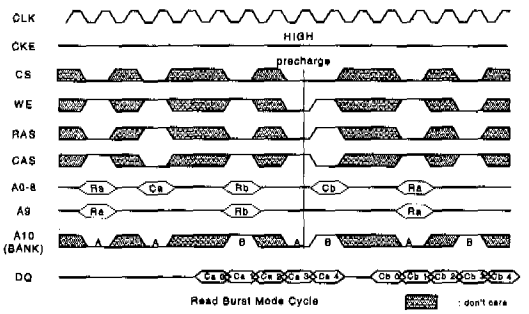


그림 3. 프레임메모리 인터페이스 모듈의 블록도



(a)



(b)

그림 4. DRAM과 SDRAM의 read cycle 비교
(a) DRAM (b) SDRAM

본 설계에서는 하드웨어 구현상의 편의를 위해서 한 매크로블록 당 27 MHz와 54 MHz 클럭에 대해서 각각 660 및 1320개의 클럭을 할당하였다. 프레임메모리 인터페이스를 위해서 54 MHz 클럭을 사용하므로, 한 매크로블록 당 1320개의 클럭을 할당하여 메모리 액세스 시간을 스케줄링 하였다.

기존에는 외부에 일반 DRAM을 사용하여 27 MHz 클럭에 동작하도록 하였기 때문에 64비트의 인터페이스 버스를 사용한 반면에, 개선된 프레임메모리 인터페이스에서는 32비트의 버스 폭을 갖는 SDRAM을 사용하고 액세스 사이클은 54 MHz로 하였다. 이렇게 설계하였을 경우에 동작속도의 상승으로 인하여 전력소모가 늘어날 것으로 여겨지나 실제 27 MHz 클럭을 사용할 경우와 비교해서 처리해야 할 데이터의 량은 동일하므로 추가적인 전력 소모는 없고, 오히려 칩 외부와의 인터페이스로 인한 전력소모가 내부 전력소모에 비해서 월등히 심각한 문제이므로 소비전력은 줄었다고 볼 수 있다[4, 5].

본 설계에 사용된 SDRAM은 256k x 32-비트 x 2-뱅크이며, 듀얼 뱅크 동작과 버스트 길이 변화를 사용하여 연속적으로 데이터를 액세스 할 때 제어가 간편해지는 SDRAM의 특징을 최대한으로 이용하였다. 그림 4에서 알 수 있듯이 DRAM의 경우, 데이터를 읽고 쓸 때마다 CAS 신호와 주소가 변경되어야 하지만, SDRAM에서는 CAS 신호와 주소의 변경을 한번만 수행하더라도 한 행의 모든 데이터를 액세스할 수 있다[6]. 듀얼 뱅크의 특징을 잘 활용함으로써, 뱅크간에 연속적인 액세스 동작의 랜덤 액세스 지연시간을 숨기게 하였다. 54MHz의 클럭을 사용하는 경우, 읽기 랜덤 액세스 시간은 4 사이클이며, 쓰는 것은 2 사이클이다. 하지만 그림 4(b)에서 보이는 것과 같은 듀얼 뱅크의 동작을 잘 활용하면, 다음 뱅크로 전환하기 위한 프리차지 한 사이클만을 사용함으로써 액세스가 가능하다. 따라서, 메모리 맵 설계는 듀얼 뱅크 동작의 사용을 최대한 이용할 수 있도록 설계하여, 각 동작에 할당된 시간을 최소화하였다.

1. 메모리 맵

그림 5는 본 논문에서 제안한 듀얼 뱅크 SDRAM의 메모리 맵이다. 이 SDRAM은 32 비트의 데이터 폭을 지니며 한 주소 당 4개의 픽셀 값을 저장한다. 순차적으로 입력되는 원 영상을 저장하는 Original Y, Original CbCr 영역에는 입력 모

듈에서 들어오는 4:2:2 영상을 저장한다. 이 영역은 실제 영상의 한 라인을 메모리의 한 행에 저장하도록 구성한 것으로, 메모리의 주소를 쉽게 지정할 수 있는 이점이 있다. 그리고 짝수 라인은 B 뱅크에, 홀수 라인은 A 뱅크에 저장함으로써, 듀얼 뱅크 동작을 최대한 활용할 수 있도록 하였다. 결국 원 영상 데이터는 라인 단위로 순차적으로 저장되며, 실제 부호화 과정에서 쉽게 활용할 수 있도록 매크로 블록 단위로 읽어 입력 모듈로 전달한다. Coded Y와 Coded CbCr영역은 부호화 된 영상의 저장 공간으로 복원된 영상을 저장한다.

coded Y영역은 180열부터 243 열까지 위치하며, 메모리 맵에서 한 라인은 한 매크로 블록을 나타낸다. Coded CbCr영역은 0 열부터 180 열까지 위치하며, Original CbCr영역과 같이 메모리맵 상의 배치가 화면위치와 동일하다. Coded Y와 Coded CbCr영역은 각각 3 개 필드 분을 사용하며, 듀얼 뱅크 동작을 최대한 이용할 수 있다. 이 영역에 대한 데이터는 제어 모듈에서 이전 영상과의 차 영상을 생성하기 위해서 사용하고, 움직임추정 및 보상 모듈에서 움직임 탐색 영역 데이터로 활용한다. VLC모듈이 발생시킨 비트스트림은 채널 버퍼에 저장되었다가 채널의 요구에 따라서 출력하도록 한다.

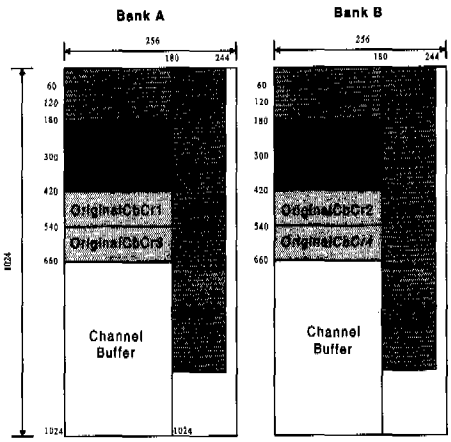


그림 5. SDRAM의 제안한 메모리 맵

2. 액세스 타이밍 스케줄

MPEG-2 비디오 인코더와 SDRAM의 인터페이스를 위한 메모리 액세스 타이밍 스케줄링 결과를 표 1에 요약하였다. SDRAM 액세스 타이밍 스케줄링의 목적은 내부 버퍼 크기와 제어 로직을 최소화하는 것이다. 내부 버퍼를 최소화시키기 위해 프레임메모리 인터페이스를 4종류로 세분화하여 액세스

표 1. SDRAM의 액세스 타이밍 스케줄

Start	End	# of clocks	Data types	# of data	Interfaces	Burst Length
1	68	68	Coded Y output	64	FM → MEMC	256
69	114	46	Original C input	42	IP → FM	256
115	167	53	Vstrm (video stream) input	49	VLC → FM	256
168	303	136	Coded Y output	128	FM → MEMC	256
304	349	46	Original Y input	42	IP → FM	256
350	402	53	Vstrm (video stream) input	49	VLC → FM	256
403	538	136	Coded Y output	128	FM → MEMC	256
539	591	53	Vstrm (video stream) input	49	VLC → FM	256
592	685	94	Original C output	88	FM → IP	4
686	695	10	Cstrm (channel stream) output	6	FM → EXT	256
696	705	10	Refresh			
706	751	46	Original C input	42	IP → FM	256
752	804	53	Vstrm (video stream) input	49	VLC → FM	256
805	850	46	Original Y input	42	IP → FM	256
851	940	90	Coded C output	50	FM → MEMC	256
941	957	17	No Operation			
958	1010	53	Vstrm (video stream) input	49	VLC → FM	256
1011	1080	70	Original Y output	64	FM → IP	4
1081	1189	109	Coded YC input	96	CTL → FM	256
1190	1199	10	Cstrm (channel stream) output	6	FM → EXT	256
1200	1252	53	Vstrm (video stream) input	49	VLC → FM	256
1253	1320	68	Coded Y output	64	FM → MEMC	256

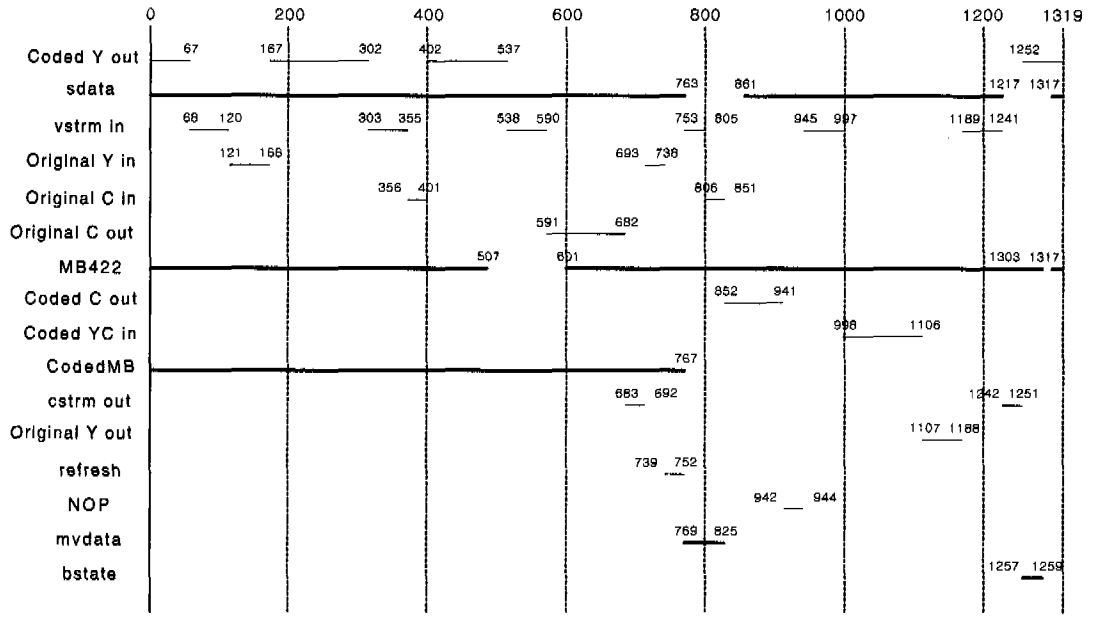


그림 7. 한 매크로블록 내의 메모리 액세스 시간

시간을 나누었다. 액세스 시간의 나누는 것은 내부 버퍼 크기를 줄이지만 전체 액세스 시간을 증가시킨다. 이는 액세스 회수를 나눈 만큼 랜덤 액세스가 늘어나기 때문이다. 따라서 내부 버퍼의 크기를 줄이기 위해서는 메모리 액세스 사이클 수를 줄이는 것이 중요하다. 표 1의 각 동작을 수행하는 클럭 수를 최소화하기 위해, 듀얼 뱅크 동작과 버스트 길이 변화를 적용하였다. 256 버스트 모드의 경우에 그림 4(b)에서 알 수 있듯이 한 뱅크에서 다른 뱅크로 주소 천이가 일어날 때 프리차지(precharge)가 필요하다. 그러나 프리차지는 뱅크 천이 시에 한 클럭을 필요로 한다. 따라서 뱅크 천이가 일어 나는 동작에서는 많은 사이클이 소비된다. 메모리 맵의 조사에 의해 버스트 길이 4의 오토 프리차지(auto precharge) 방법을 입력 모듈 인터페이스에 적용할 수 있다. 오토 프리차지는 한 번 정의해 준 버스트 길이 만큼 읽기 및 쓰기를 수행하면 자동적으로 뱅크가 바뀌어 별도로 프리차지 사이클을 필요로 하지 않는다. 따라서 뱅크의 변화가 많은 작업에는 기본 사이클 수를 줄일 수가 있다. 그림 6은 버스트 길이를 4 로 설정하고, 버스트 길이 4로 읽어 내며, 다시 버스트 길이를 256으로 설정하는 과정을 보여주고 있다.

그림 4(b)에는 프리차지 사이클이 존재 하지만, 그림 6에서는 존재하지 않는다. 따라서 프리차지 사이클 대신에 오토 프리차지 방법을 적용함으로써 기본 사이클 수를 줄일 수 있으며, 여기서 절약된 사이클을 액세스 시간의 분할에 사용함으로써 내부 버퍼의 크기를 줄일 수 있다. 프레임메모리 인터페이스 모듈내의 시퀀스 디코더는 각 서브 모듈의 SDRAM액세스 시간을 제어한다. 액세스 타이밍은 외부 모듈로 나가는 데이터의 시간을 고정시킨 상태에서 SDRAM의 액세스 시간을 배치하여 스케줄링 하였다. 그림 7은 한 매크로블록 시간에 대한 스케줄링 결과이다. 여기서 가는 실 선은 각 서브 모듈의 SDRAM 액세스 시간이고 굵은 실 선은 다른 모듈과의 액세스 시간이다.

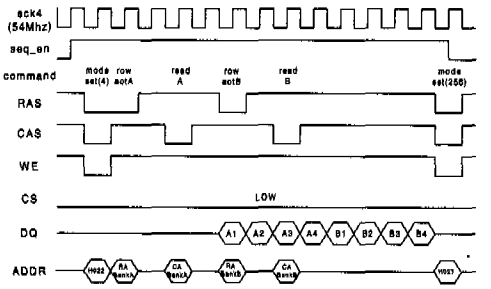


그림 6. 버스트 길이 변화 및 오토 프리차지

Ⅳ. 설계 및 시뮬레이션 결과

표 2. 기존의 구조와 제안한 구조의 내부 SRAM 크기 비교표

Sub Module	Input / Output	Previous design			Proposed design		
		Buffer size(byte)	Hardware area(mil^2)		Buffer size(byte)	Hardware area(mil^2)	
IP	Input	1024(=512+512)	Y	2559.152	512(=256+256)	Y	972.44
			C	2559.152		C	972.44
	Output	768(=256+384)	Y	2695.81	768(=256+384)	Y	972.44
			C	3108.07		C	1276.14
MEMC	Output	768(=256x3)	5275.67(=20.13x10.92x24)		768(=256x3)	2209.11(=60.74x36.37)	
CTL	Input	512(=256+256)	1267.6(=20.13x13.46x8)		384(=256+128)	1588.6(=62.03x25.61)	
VLC	Input	2048(=32x256x2)	4263.95(=60.74x35.10x2)		768(=32x192)	2209.11(=60.74x36.37)	
	Output	128(=8x16x8)	1347.90(=20.13x8.37x8)		64(=32x16)	508.39(=60.74x8.37)	
Total		5248	23077.304		3264	10708.67	

표 3. 개선 결과

Designs	Buffer size (Kbyte)	Area (mm^2)	Number of Tr.
Previous design[3]	5.248	46	259,972
Improved design	3.264	19.5	197,228

본 논문에서는 SDRAM의 특성을 이용하여, 듀얼 뱅크 동작과 버스트 모드의 적절한 선택으로 각 동작 당 필요한 클럭 수를 줄임으로써 그 여유 분의 클럭을 여러 번의 랜덤 액세스 시간에 할당하여, 버퍼 크기를 줄일 수 있었다. 또한 데이터 버스 폭을 32 비트로 줄여 외부 인터페이스를 간편하게 하였다. 그 결과 표 2 에 보이는 것과 같이 기존의 구조에 비해서 버퍼의 크기는 약 38% 줄어들었으며, 사용된 버퍼가 차지하는 하드웨어 면적은 54% 까지 줄었다. 이 이유는 기존의 구조가 single-port 메모리(SRAM)를 사용하는 반면에 제안한 구조는 dual-port 메모리를 사용하기 때문이다[7]. 이는 메모리 타입의 선택이 하드웨어 설계의 효율에 대단한 영향을 미친다는 것을 의미한다. 트랜지스터 수에 대해서도 표 3에서 알 수 있듯이 24% 이상 절약되었다. 기존의 비디오 인코더에 대한 전체 하드웨어 면적은 108.94 mm2였으며, 이 중에서 46 mm2가 프레임메모리 인터페이스를 위해서 소모되었다. 즉 전체 면적의 약 42%를 프레임메모리 인터페이스 모듈이 차지하였다. 본 논문에서 개선한 구조를 레이아웃한 결과 19.5 mm2의 하드웨어 면적을 차지하여, 기존 프레임메모리 인터페이스 모듈의 42%에 해당하는 면적을 차지하였다. 결국 본 구조는 전체 하드웨어 면적을 약 24.3% 절약하는 효과가 있다.

제안된 구조는 MPEG-2 비디오 인코더의 하드웨어 구조에 대한 C-코드 모델링을 통하여 생성된 테스트 벡터를 이용하여 기능이 검증되었다. 기능 검증은 그림 8과 같이 C-코드 모델링으로부터 생성된 테스트 벡터를 VHDL로 기술된 테스트 벤치에서 기준 값으로 받아들이고 하드웨어 설계 결과로부터 생성되는 실제 값과 비교함으로써 수행되었다.

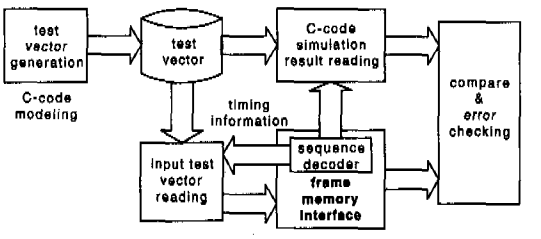


그림 8. 하드웨어 검증을 위한 테스트 벤치의 구성

V. 결 론

본 논문에서는 MPEG-2 비디오 인코더의 하드웨어 면적을 줄이기 위해서 프레임메모리 인터페이스 모듈의 효율적인 구조를 제안하였다. 제안한 구조는 64 비트를 사용하던 기존 DRAM구조에서 32 비트

의 버스를 갖는 SDRAM을 사용하였고, 효과적인 메모리 맵과 액세스 스케줄링을 통하여 내부 버퍼가 차지하는 하드웨어 영역을 54% 가량 줄였다. 아울러 내부 제어 로직을 간략화 하여 게이트 수를 24% 이상 줄여, 추가적인 하드웨어 면적 감소가 가능하였으며, 외부 인터페이스 버스 폭을 줄여 전력 소모에 유리한 구조를 가지도록 하였다. 제안한 구조는 VHDL로 설계되었으며 0.5 m CMOS TLM 3.3 V 표준 셀 라이브러리를 사용하였다.

본 논문에서 제안한 하드웨어 구조 개선은 기존의 비디오 인코더 칩 면적에서 42% 가량을 차지하던 프레임 메모리 인터페이스 모듈의 면적을 58% 가량 줄임으로써 전체 칩 면적을 24.3% 가량 줄일 수 있음을 확인하였다.

참 고 문 헌

[1] ISO/IEC JTC1/SC29/WG11 13818-2 : Moving Picture Experts Group, IS, May 1996.

[2] K Kim, J. S. Yoon, S. H. Jang, J. H. Hwang, J. S. Hyun, S. H. Jang, and S. H. Kwon , A VLSI Implementation of MPEG-2 Video Encoder, *The 5th International Conference on VLSI and CAD*, October 13-15, 1997, Seoul , Korea.

[3] Y. K. Ko, K. H. Lee, E. S. Kang, S. H. Lee, S. H. Jang, S. J. Ko, The design of Frame memory module for MPEG-2 video encoder, *The Summer Conference of KICS*, vol 15, No.1, pp. 452-458, July 1996.

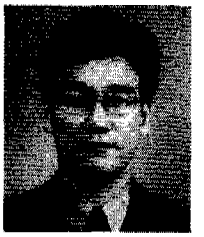
[4] A. P. Chandrakasan, and R. W. Brodersen, Minimizing Power Consumption in Digital CMOS Circuits, *Proceedings of the IEEE*, vol. 83, No. 4, pp. 498-523, April 1995.

[5] T. H. Meng, Low-Power Wireless Multimedia Applications, *IEEE Communications Magazine*, pp.130-136, June 1998.

[6] Samsung Electronics Co., Ltd., *Graphic Memory*, pp. 231-278, May, 1997.

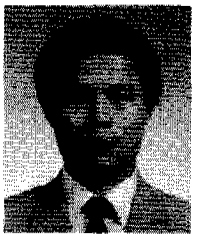
[7] VLSI Technology Inc., *0.5-micron HDI 3V core cell-based libraries*, Sept. 1995.

김 견 수(Kyeounsoo Kim) 정회원



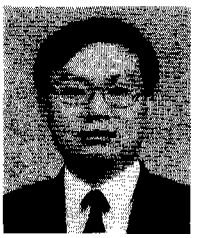
1986년 2월 : 동아대학교 전자공학과 졸업
1988년 8월 : 부산대학교 산업대학원 석사
1997년 2월 : 부산대학교 전자공학과 박사
1990년~현재 : 한국통신 연구개발본부 가입자망연구소 전임연구원
1998년 12월~현재:과학재단 지원 Post-Doc. EE-Systems Dept., USC
<주관심 분야> VLSI 신호처리, 소스/채널코딩

고 종 석(Jong-Seog Koh) 정회원



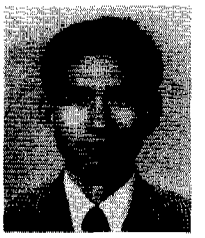
1982년 2월 : 고려대학교 전자공학과 졸업
1984년 2월 : 한국과학기술원 전기및전자공학과 석사
1989년 2월 : 한국과학기술원 전기및전자공학과 박사
1989년~현재 : 한국통신 연구개발본부 가입자망연구소 책임연구원 (실장)
<주관심분야> 영상통신, 소스/채널코딩, IMT-2000

서 기 범(Ki-Bum Suh) 정회원



1989년 2월 : 한양대학교 전자공학과 졸업
1991년 2월 : 한양대학교 전자공학과 석사
1991년~현재 : 한양대학교 전자공학과 박사과정
<주관심분야> 영상통신 알고리즘 및 아키텍처

정 정 화(Jong-Wha Chong) 정회원



1975년 2월 : 한양대학교 전자공학과 졸업
1977년 2월 : 한양대학교 전자공학과 석사
1981년 : 와세다대학교 전자공학과 박사
1979년 3월~1980년12월 : NEC 중앙연구소 근무
1981년 3월~현재 : 한양대학교 전자공학과 교수
<주관심 분야> VLSI 설계, 영상압축, 디지털통신