논문 2007-44SD-12-14

모바일 환경에서의 H.264 / AVC를 위한 인트라 예측기의 구현 및 검증

(Implementation and verification of H.264 / AVC Intra Predictor for mobile environment)

윤 철 환*, 정 용 진**

(Cheol-Hwan Yun and Yong-Jin Jeong)

요 약

작은 면적과 저전력으로의 구현은 다양한 멀티미디어 하드웨어, 특히 모바일 환경에서 매우 중요한 요구사항이다. 본 논문은 작은 면적과 그에 따른 저전력을 목표로 H.264/AVC 인트라 예측기의 하드웨어 구조를 제안한다. 이미지 프레임을 예측하기 위해 하나의 연산기로 모든 모드 결정과 계산들이 순차적으로 수행되고 그들 중 최적의 값을 선택하는 방식이며, 그 결과로 다른 기존의 논문들 보다 더 작은 면적의 결과를 얻을 수 있었다. 제안된 구조는 Altera Excalibur device를 이용하여 검증되었고, 구현된 하드웨어 구조는 Synopsys Design Compiler와 Samsung STD130 0.18um CMOS Standard Cell Library를 이용하여 합성하였다. 합성결과 크기는 11.9k의 하드웨어 로직 게이트와 1078 byte의 내부 SRAM을 사용하고 최대 동작 주파수는 약 107MHz가 되었다. 제안한 구조는 하나의 QCIF(176x144 화소) 영상 프레임을 처리하는데 879,617클록이 소요되며, 이는 QCIF 영상을 초당 121.5프레임으로 처리가 가능하며, 이는 하드웨어 기반의 실시간 H.264/AVC 부호화 시스템에 적합한 구조임을 보여준다.

Abstract

Small area and low power implementation are important requirements for various multimedia processing hardware, especially for mobile environment. This paper presents a hardware architecture of H.264/AVC Intra Prediction module aiming on small area and low power. A single arithmetic unit was shared and processed sequentially for all mode decisions and computations to predict an image frame. As a result, we could get smaller area and smaller memory size compared to other existing implementations. The proposed architecture was verified using the Altera Excalibur device, and the implemented hardware has been described in Verilog-HDL and synthesized on Samsung STD130 0.18um CMOS Standard Cell Library using Synopsys Design Compiler. The synthesis result was about 11.9K logic gates and 1078 byte internal SRAM and the maximum operating frequency was 107Mhz. It consumes 879,617 clocks to process one QCIF frame, which means it can process 121.5 QCIF(176x144) frames per second, therefore it shows that it can be used for real time H.264/AVC encoding of various multimedia applications.

Keywords : H.264 / AVC, Intra Prediction, small area, single arithmetic unit

I. 서 론

디지털 멀티미디어 방송을 위한 국내 지상파 및 위성파 DMB 표준에서는 ITU-T/MPEG에서 제정한 H.264/AVC를 동영상 부호화 기법으로 채택하였다. 이 H.264/AVC 표준 은 다양한 서비스를 제공하기 위하여 동영상 처리를 VCL(Video Coding Layer)와 NAL(

* 학생회원, ** 정회원, 광운대학교 전자통신공학과
(Dept. of Electronics and Communications Engineering, Kwangwoon University)

※ 본 논문은 IDEC/IT-SoC 사업단의 툴지원과 서울시 혁신 클러스터 (나노 IP-SoC) 사업 및 광운대학교 2007 교내학술 연구지원으로 이루어졌습니다.
접수일자: 2007년8월27일, 수정완료일: 2007년11월26일

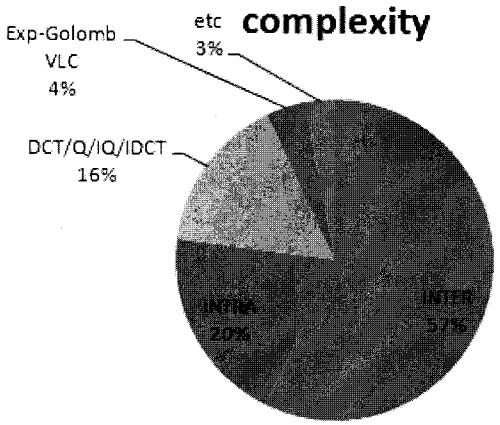


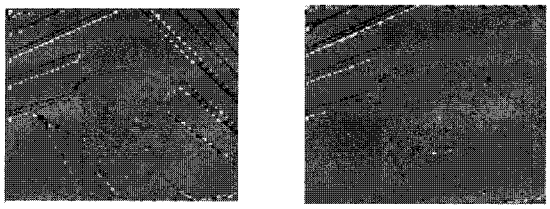
그림 1. 부호화기의 복잡도 분석[1]
Fig. 1. Analysis of the complexity of Encoder[1].

Network Abstraction Layer)로 나누었으며 , 1/4 픽셀 움직임 예측 및 보정, 인트라 예측 , 디블록킹 필터 등의 여러 기법을 적용하여 이전 표준보다 최대 60%이상의 압축 효율을 높였다. 이 중 인트라 추정은 부호화하는 슬라이스의 유사성을 제거하는 기법으로서 그림 1에서 보듯 전체 H.264/AVC의 부호화시에 약 20% 연산량을 차지하는 중요한 부호화 모듈이다.[1] 이는 데스크탑 환경의 소프트웨어 수행 결과로써 모바일 환경에서 동영상을 실시간으로 부호화하기 위해서는 하드웨어 기반의 처리가 요구된다. 본 논문에서는 작은 면적을 요구하는 모바일 환경에서 동영상을 실시간으로 처리하기 위한 작은 하드웨어 사이즈를 가지는 인트라 예측기 하드웨어를 Altera Excalibur 기반으로 구현하고 검증하였다.

본 논문에서는 적은 게이트 수 구현을 위해 하나의 예측 연산기를 반복해서 사용하는 구조를 제안하였다. 본 논문의 구성은 먼저 Ⅱ장에서는 인트라 예측기의 알고리즘을 설명하고 Ⅲ장에서는 제안한 하드웨어 구조를 설명한다. Ⅳ장에서는 구현된 전체 하드웨어 구조의 결과 및 성능을 분석하며 마지막으로 Ⅴ장에서 결론을 맺는다.

Ⅱ. 인트라 예측기의 알고리즘

인트라 예측은 현재 슬라이스의 공간적 중복성을 제거하기 위하여 부호화된 이전 매크로블록의 픽셀 값을 참조하여 가장 유사한 매크로블록을 생성하는 것이다. 인트라 예측은 휘도 블록의 인트라 4화소x4화소 단위 예측과 16화소x16화소 단위의 예측 그리고 채도 블록의 8화소x8화소 예측으로 나누어진다.[2]



(a) (b)

그림 2. 차분 영상의 비교 : (a) 수평, 수직, DC모드로만 예측; (b) 9가지 모드 모두 예측
Fig. 2. Comparison of residual image; (a) Only vertical, horizontal, DC prediction; (b) All 9 modes prediction

H.264에서는 수직, 수평, 평균값만을 이용하는 다른 코덱과는 달리 다양한 방향의 모드를 가지고 있어서 좀 더 차분영상의 값을 작게 할 수 있는 예측값을 산출한다. 그림 2에서 보듯 다양한 인트라 예측 모드로 예측을 수행했을 때 회색에 가까운 값들이 나오는데 이는 차분 영상이 0에 가깝다는 것을 나타낸다. 이는 영상의 공간주파수가 낮아져서 변환이 용이한 형태가 된다.[2]

본 장에서는 위에서 언급한 모드에 따른 알고리즘에 대하여 알아보고 모드를 결정하는 방법에 대하여 설명한다.

1. 인트라 예측 모드

H.264/AVC 인트라 예측에서 휘도 4x4 모드는 그림 3과 같이 주변의 이미 코딩된 픽셀들을 이용해서 8가지의 방향으로 예측을 한다. 주변의 코딩된 픽셀의 평균값을 내는 DC 모드까지 합쳐서 총 9가지의 모드로 결정을 한다.

또한, 휘도 인트라 예측에는 16x16으로 매크로 블록 전체를 코딩하는 방법이 있다. 이 방법은 구름 한 점 없는 푸른 하늘같은 공간적 색의 유사도가 높은 영상이나 영상의 크기가 큰 영상에서 주로 선택 된다. 휘도 16x16 모드에는 4가지가 있는데 방법은 그림 4와 같다.[2]

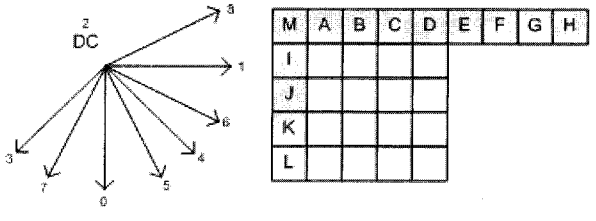


그림 3. 인트라 4화소x4화소 예측모드의 방향과 참조픽셀[2]
Fig. 3. Direction of Luma 4x4 mode and Ref. Pixels.[2]

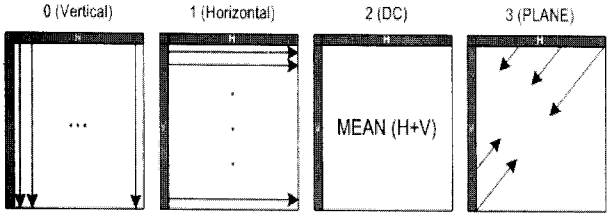


그림 4. 인트라 16화소x16화소 예측모드[2]
Fig. 4. Prediction modes of Luma 16x16.[2]

2. 모드 결정

H.264 JVT(Joint Video Team) 표준화 활동에서 규격서의 작성이 진행되는 동시에 독일의 연구기관 HHI (Heinrich-Hertz-Institute)를 중심으로 참조 소프트웨어의 개발이 진행되었다. 이것을 JM (Joint Model) 이라고 불리는데 이 모델은 인트라와 인터 예측에 대하여 고화질 모드(High Complexity)와 고속 모드(Low Complexity)라는 두 가지 판정 모드로 구성되어 있다.[2] 인트라 예측의 경우 최적의 모드를 결정하기 위한 비용 산출 식은 각 경우에 따라 다음과 같다.[2]

$$Cost_h = SSD + \lambda_{\mathrm{mode}} \cdot GenBit \quad (1)$$

$$Cost_l = SA(T)D + QP2Quant(QP) \times HeaderBit \quad (2)$$

$Cost_h$: 고화질 모드의 비용함수

$Cost_l$: 고속 모드의 비용함수

λ_{mode}: 모드판정에 대한 라그랑제 승수

$GenBit$: 매크로블록을 해당 부호모드로 부호화시 발생비트

$QP2Quant(QP)$:양자화 파라미터를 이용한 양자화 스케일의 변환값

$HeaderBit$:헤더비트 부분을 비용으로 고려한 값

$$SAD = \sum_{i,j} |Diff(i,j)| \quad (3)$$

$$SSD = \sum_{i,j} [s(i,j) - c(i,j)]^2 \quad (4)$$

$$SATD = (\sum_{i,j} |DiffT(i,j)|)/2 \quad (5)$$

$Diff(i,j) = Original(i,j) - \mathrm{Prediction}(i,j)$

$s(i,j) = OrigianlBlock$

$c(i,j) = ReconstructBlock$

$DiffT = (H^* Diff^* H)/2$

$H =$ Hadamard Transform Coefficient[2]

고화질 모드로 예측이 이루어졌을 경우 일반적으로 고속모드보다 부호화 효율이 10%전후 높다고 알려져 있다.[2] 하지만 제곱 연산과 같은 복잡도가 높은 연산식을 포함하고 있기 때문에 모든 픽셀의 예측 연산을 수행했을 시에 실시간 처리가 어렵다는 단점이 있다. 반면에 고속 모드는 덧셈, 뺄셈과 절대값 연산만을 이용하기 때문에 고화질 모드보다 복잡도가 낮아 하드웨어 구현에 더 적합하다.

고속 모드에서는 SAD (Sum of Absolute Difference) 혹은 SATD (Sum of Absolute Transformed Difference)를 이용해 비용을 산출한다. 이때, SATD보다 SAD를 사용했을 때 계산이 단순해지지만 실험적으로 PSNR이 0.3dB정도 낮다고 알려져 있다.[3] 본 논문에서는 작은 면적과 저전력의 코아 설계를 위해 고속 모드의 SAD연산을 이용한 구조로 설계를 하였다.

3. 인트라 예측 모드

전술한대로 H.264는 다른 코덱보다 많은 인트라 모드를 가지고 있다. 인트라 모드가 많으면 많을수록 차분 영상은 0에 더 가깝게 나오겠지만 디코더에 알릴 모드의 정보를 나타내는 비트수는 늘어나게 되는데 이것은 인접하고 있는 4x4블록의 모드 간에 존재하는 서로 유사성을 이용하여 줄일 수 있다.

그림 5의 예에서 현재 블록의 상단 블록 A가 1번 수평 모드로 선택이 되었고, 좌측의 블록 B가 2번 DC 모드로 선택이 되었다고 가정하면 두 모드 중의 더 작은 모드인 1번 수평(horizontal) 모드가 현재의 가장 가능성 있는 모드(most probable mode)라고 규정한다. 현재의 블록 E가 1번 수평 모드로 결정 되었다면 예측 모드 플래그(prev_intra4x4_pred_mode)는 true가 되고 더 이상의 모드 정보는 보내지 않는다. 만약에 1번 모드가 아닌 다른 모드로 결정 되었다면 위 신호는 false가 되

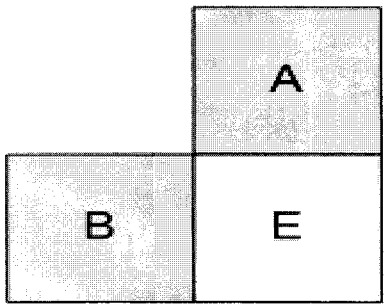


그림 5. 인접 4x4 블록의 모드 참조
Fig. 5. Reference of neighbor 4x4 block.

표 1. 예측 모드의 선택
(가장 가능성 있는 모드 = 1)
Table 1. Selection of prediction mode.

selected mode	prev_intra	rem_intra
0 (vertical)	false	000
1 (horizontal)	true	X
2 (DC)	false	001
3 (DDL)	false	010
4 (DDR)	false	011
5 (VR)	false	100
6 (HD)	false	101
7 (VL)	false	110
8 (HU)	false	111

고 예측 모드의 부호화 데이터(rem_intra_4x4_pred_mode)를 보내게 된다. 표 1에서 보듯 이 신호는 3-bit의 신호로 예측한 모드 보다 작으면 선택된 모드가 그대로 입력되고 크다면 선택된 모드에 1을 뺀 값이 새로 부호화 되어 전송이 된다.[4]

4. 인트라 평면 예측

휘도 16x16 평면 예측 (3번 모드)은 식 4~9에서 보듯 복잡한 계산이 요구된다.

$$P[y,x] = Clip((a+b\cdot(x-7)+c\cdot(y-7)+16) \gg 5)$$
(한 매크로블록 x,y 좌표에서의 픽셀값) (6)

$$a = 16\cdot(p[-1,15]+p[15,-1] \quad (7)$$

$$b = (5\cdot H+32) \gg 6 \quad (8)$$

$$c = (5\cdot V+32) \gg 6 \quad (9)$$

$$H = \sum_{x}^{7}(x'+1)\cdot(p[-1,8+x']-p[-1,6-x']) \quad (10)$$

$$V = \sum_{y}^{7}(y'+1)\cdot(p[-1,8+y']-p[-1,6-y']) \quad (11)$$

휘도 16x16 mode 3의 예측값을 구하기 위해서는 곱셈 2번 덧셈 3번 쉬프트 1번을 요구한지만 중복되는 연산이 있으므로 다음과 같은 방법으로 계산을 간략화 할 수 있다.

$$A_{00} = a+b\times(-7)+c\times(-7)+16$$
$$P[0,0] = Clip(A_{00} >> 5)$$

$$A_{01} = A_{00}+b$$
$$P[0,1] = Clip(A_{01} >> 5) \quad (12)$$

$$B_{00} = A_{00}+c$$
$$P[1,0] = Clip(B_{00} >> 5)$$
$$B_{01} = B_{00}+b$$
$$P[1,1] = Clip(B_{01} >> 5) \quad (13)$$

휘도 16x16 평면 예측 모드에서 이렇게 변환된 방법으로 연산을 수행한다면 곱셈 2번 덧셈 3번의 계산이 덧셈 1번만으로 계산을 끝낼 수가 있다. 이것은 연산량이 줄어들었기 때문에 수행시간을 줄이는 것과 추가적인 연산기를 두지 않아도 되기 때문에 하드웨어 설계에 적합한 계산 방식이다.

III. 인트라 추정기의 구조

본 논문에서 제안된 인트라 예측기의 구조는 H.264의 전체 모듈 구성을 할 때 인트라 예측과 인터 예측이 동시에 수행되므로 인터 예측기와 수행 성능을 맞추는 방향으로 설계되었다. 본 논문이 참조한 [5]의 인터 예측기의 성능은 하나의 매크로블록 당 7,146클록에 15.99ns의 최장 지연시간을 가진다. 그림 1에서 보듯 인터 예측은 인트라 예측보다 복잡도가 높다. 그러므로 본 논문에서의 인트라 예측기는 속도의 성능 보다는 하드웨어의 크기를 줄이고 또한 그에 따른 전력 소모를 줄이는데 목표를 두고 구현되었다.

1. 내부 메모리

매크로 블록 당 많은 모드를 처리 하고 빠른 수행 시간을 위해서는 내부 메모리의 크기를 적절하게 책정하는 것이 무엇보다 필요하다. 본 논문에서는 매크로블록 단위로 데이터를 내부 메모리에 저장하여 처리하는 방식으로 부호화의 효율을 높였다.

1.1 수평 버퍼 & 수직 버퍼

수평 버퍼 (Row buffer)는 매크로 블록을 코딩할 때 수평으로 참조되는 데이터들을 저장하기 위한 메모리이고, 수직 버퍼 (Column buffer)는 수직으로 참조되는 데이터들을 저장하기 위한 메모리이다. 수평 버퍼의 구성은 그림 6과 같이 한 라인의 크기를 가지고 수직 버퍼의 구성은 매크로 블록의 수직 크기만큼의 크기를 가진다. 두 경우 모두 하나의 매크로 블록의 부호화가 완

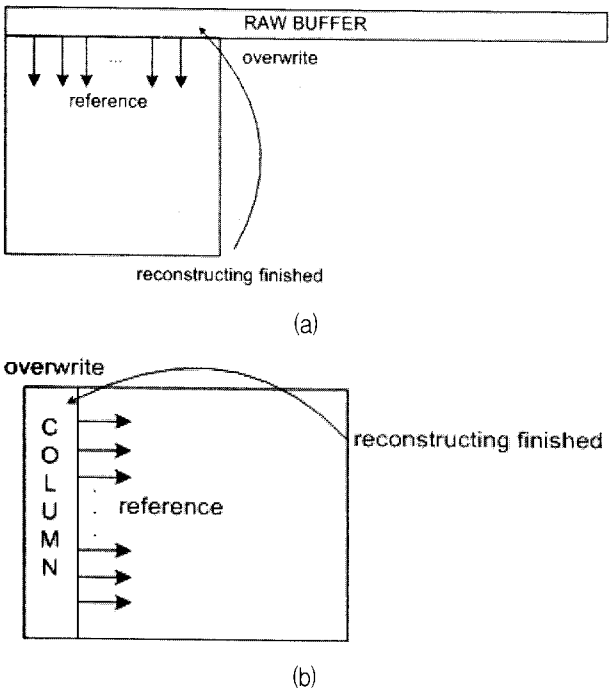


그림 6. (a) 수평 버퍼의 업데이트
(b) 수직 버퍼의 업데이트
Fig. 6. (a) Update of Row Buffer.
(b) Update of Column Buffer.

료 될 때 마다 데이터를 덮어쓰는 형식으로 업데이트 된다.

1.2 에지 버퍼

수평 버퍼와 수직 버퍼를 위와 같은 방법으로 업데이트 할 경우에는 하단의 매크로 블록을 처리할 때 필요한 에지 부분의 참조 데이터가 손실된다. 그림 7과 같이 손실을 막기 위해 각 버퍼들을 업데이트하기 전에 에지 버퍼 (Edge buffer)에 코딩된 우측 하단의 데이터를 저장시킨다.

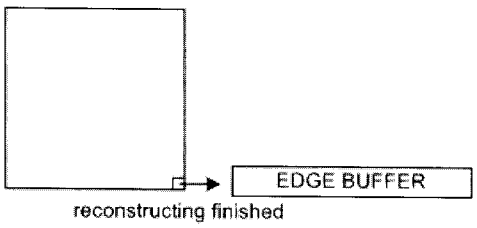


그림 7. 에지 버퍼의 업데이트
Fig. 7. Update of Edge Buffer.

1.3 블록 참조 레지스터

4x4블록을 코딩하기 위해서는 매 블록마다 13개의 주변 픽셀들을 필요로 한다. 이 데이터들을 레지스터화 시켜 4x4블록의 처리를 고속화 하였다.

1.4 내부 메모리의 전체 크기

위에서 열거한 메모리 외에도 매크로 블록 크기의 차분 영상과 코딩된 영상을 저장하는 메모리와 예측 모드를 계산하기 위해 쓰이는 전 모드 메모리가 내부 메모리로 쓰여 졌다. 인트라 예측기 모듈을 계산 하는데 사용된 총 메모리 크기는 좌측 하단의 표 2와 같다.

표 2. 전체 내부 메모리 크기
Table 2. Total internal memory size.

	SIZE	TOTAL
Reconstruct	16x16x8bit	2048 bit
Residue	16x16x9bit	2304 bit
Original	16x16x8bit	2048 bit
Row Buffer	16x11x8bit	1408 bit
Column Buffer	16x8bit	128 bit
Edge Buffer	22x8bit	176 bit
Mode data	(4x4x4bit) + (11x4x4bit) + (4x4bit)	256 bit
TOTAL	8,624 bit	

2. 예측 연산기 (Prediction Calculator)

인트라 예측의 주요 계산은 예측 값 생성 SAD값 산출 계수 업데이트로 나누어질 수가 있다. 그림 8과 같이 게이트 수를 줄이기 위하여 이 계산을 16x16 3번 모드인 PLANE을 제외하고 하나의 연산기에서 수행이 될 수 있도록 하였다. 휘도 4x4 모드 3~8의 경우는 13개의 전 픽셀 레지스터에서 모드와 픽셀의 위치에 맞는

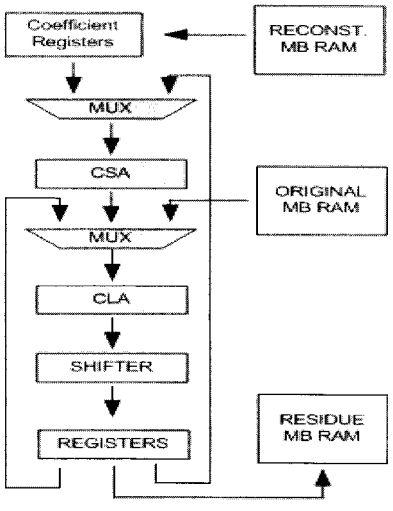


그림 8. 예측 연산기의 구조
Fig. 8. Architecture of Prediction Calculator.

데이터를 선택하여 덧셈을 한 뒤에 쉬프트 연산을 수행하고 임시 데이터 저장 레지스터에 데이터를 넣는다. 그 뒤 다음 클록에 읽어 들인 원본 픽셀 값과 차를 구한다. 다중 입력의 계산을 빠르게 수행하기 위해 CSA와 CLA pair를 이용하여 처리하였다. 한 클록에 차분값을 구할 수 있는 모드 0~2(휘도 16x16 포함)와 DC 계수 누산 같은 경우는 CSA를 거치지 않고 중간 단계부터 수행된다.

3. 평면 모드 연산기 (PLANE Calculator)

인트라 예측의 가장 복잡한 계산은 휘도 16x16 PLANE 계수들의 계산이다. 식 (8), (9)의 H, V 계산을 그림 8과 같은 구조로 구현한다면 멀티플렉서의 크기가 H, V 계산에 필요한 32개의 주변 레지스터와 b, c계산에 필요한 계수만큼 커지게 된다. 그렇게 되면 멀티플렉서의 게이트 수가 증가하게 될 뿐만 아니라 콘트롤러가 복잡하게 되고 최장 지연시간도 길어지게 된다. 그래서 휘도 16x16 PLANE 모드를 계산하는 모듈을 따로 두어서 콘트롤러가 복잡하게 되는 것과 최장 지연시간이 길어지는 것을 막았다. 그림 9에서 평면 모드 연산기의 주요 모듈인 H, V 연산기 (H, V calculator)와 b, c 연산기 (b, c calculator)를 나타내었다.

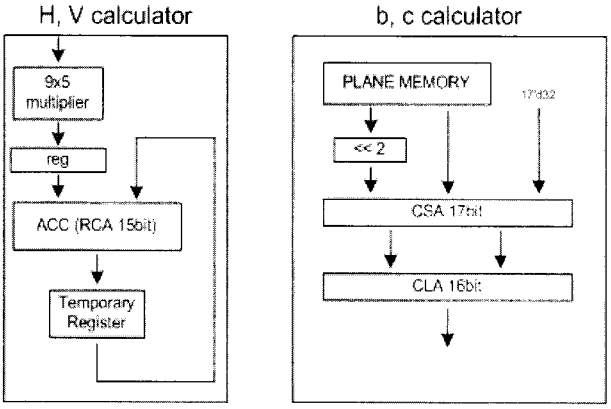


그림 9. 평면 모드 연산기의 주요 모듈
Fig. 9. Main module of PLANE Calculator.

4. Customized Adder (based on CLA)

본 인트라 예측기의 구조에서 가장 빈번하게 일어나는 연산인 SAD값을 계산은 식(3)과 같이 차분값을 구한 뒤에 절대값을 취하는 연산이다. SAD에는 두 번의 연산 동작이 필요한 데 그림 8의 구조대로 한다면 SAD 연산을 수행하기 위하여 차분값을 구하는데 덧셈기를

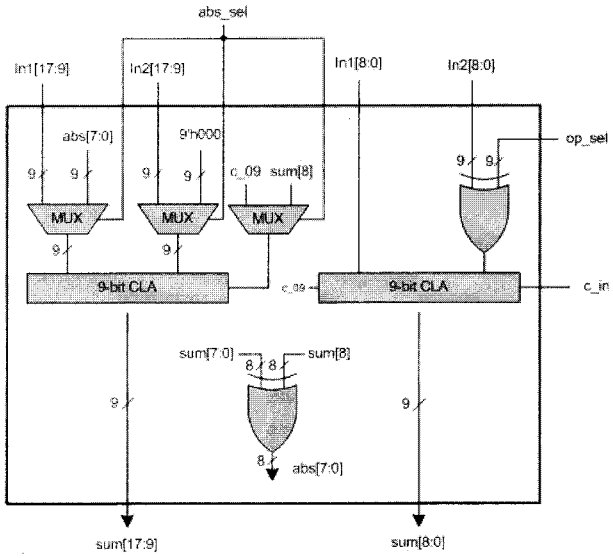


그림 10. 절대값 계산을 위해 변형된 CLA
Fig. 10. Customized Adder for calculating ABS.

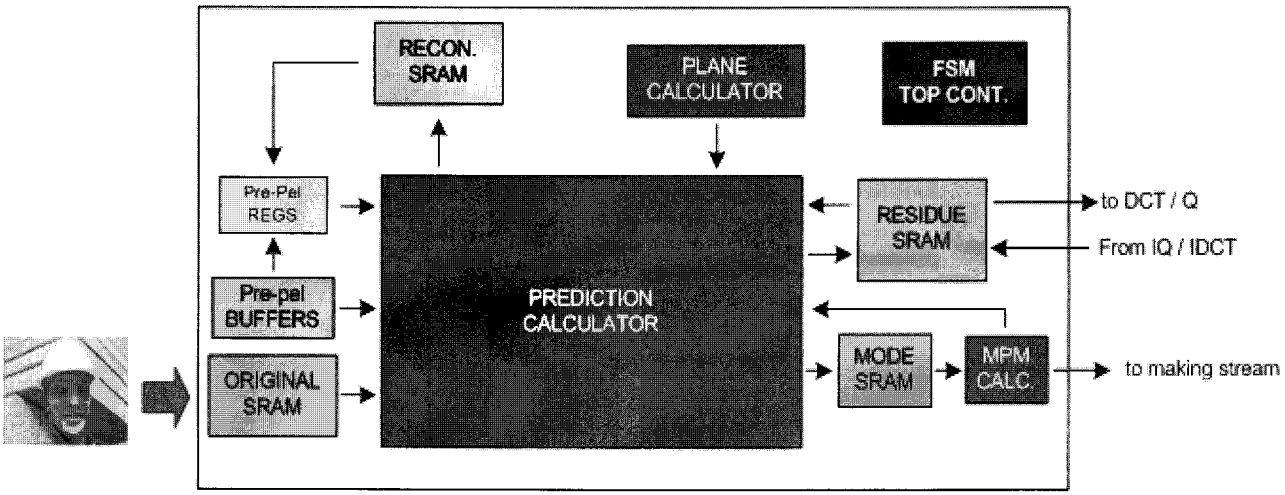


그림 12. 인트라 예측기의 전체 구조
Fig. 12. Top architecture of intra prediction module.

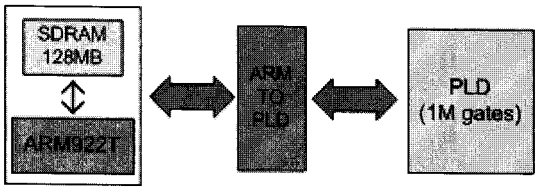


그림 11. Altera Excalibur를 이용한 검증도
Fig. 11. Verification diagram using Altra Excalibur.

한 번 거치고 크기를 구하는데 다시 한 번 거치면서 총 두 클록이 수행된다. 본 덧셈기에서 가장 데이터 패스가 큰 연산이 18-bit연산이고 SAD 연산은 8-bit의 계산이다. 이는 9-bit의 덧셈기로 SAD 연산의 차분을 구하고 9-bit 덧셈기로 절대값을 취하는 것으로 한 패스에 차분과 절대값을 모두 계산할 수 있다. 그래서 본 덧셈기의 구조는 그림 10에서와 같이 18-bit 덧셈기중 하위의 비트는 차분값을 구하고 상위비트는 절대값을 계산할 수 있도록 구조를 변형시켰다. 이 구조는 각 블록마다 수행하는 예측 모드의 수 × 한 프레임의 픽셀 수만큼의 클록을 줄이는 효과를 가져왔다. 이것은 총 수행시간의 약 40%를 절약 하는 효과이다.

지금까지 열거된 모듈들을 통합한 전체 구조가 뒷장의 그림 12에서 보여 지고 있다.

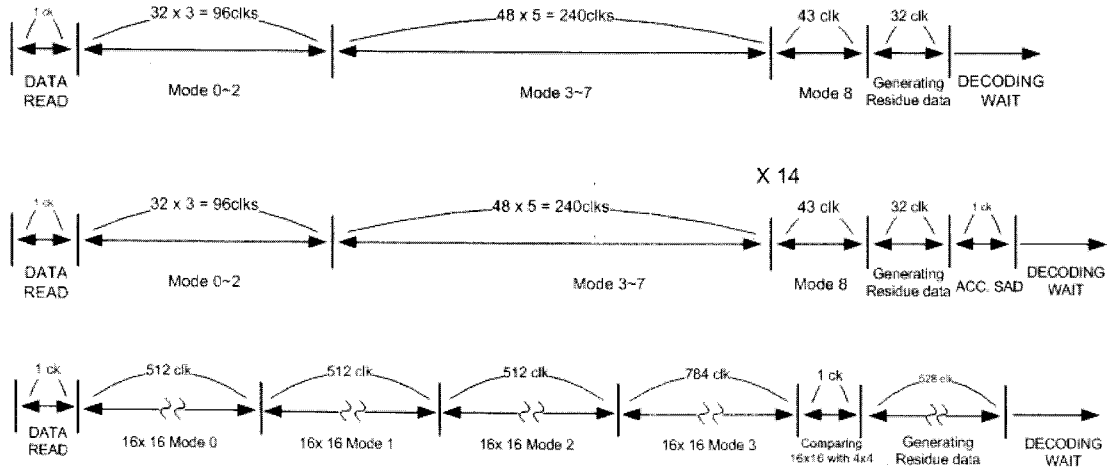
IV. 결과 및 성능 분석

제안한 인트라 예측기 하드웨어 구조의 기능을 검증하기 위해 적합한 테스트 벡터를 선택하여, 이를 올바른 기능적 동작이 이루어지는지 체크하는 시뮬레이션 과정과 SoC(System-on-chip)환경을 구축한 후, 설계한 모듈을 추가하여 실제 동작을 확인하는 에뮬레이션을 과정을 거쳐야 한다.

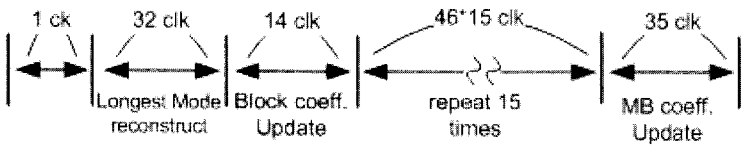
먼저 원본 영상 데이터 중 임의의 프레임을 실제 테스트 벡터로 선택하여 동작의 정확성을 높이도록 하였다. 또한 디코딩된 영상을 추출하기 위하여 [6]에서 역이산 여현 변환(Inverse DCT)된 차분 영상을 테스트 벡터로 사용하였다.

기능적으로 검증된 인트라 예측기를 에뮬레이션하기 위하여 Altera사의 FPGA 디바이스인 Excalibur를 사용하였다. Excalibur는 ARM922T와 APEX20KE가 온칩 버스로 연결된 SoC(System-On-Chip) 디바이스로 소프트웨어 하드웨어 통합 검증을 제공한다. 그리고 ARM과의 PLD 영역의 통신을 위하여 ARM사의 AMBA 버스와 연결할 수 있도록 AHB-TO-PLD 브릿지를 설계하였다[7].

합성한 결과 인트라 예측기는 EPXA10F1020C2 디바이스 기준으로 최대 동작 주파수는 25.8MHz이며 약



(a)



(b)

그림 13. 소요된 클록의 분포도 : (a) 부호화기; (b) 복호화기 (또는 복호화 루프)
Fig. 13. Distribution chart of consumed clocks : (a) encoder; (b) decoder (or decoding loop)

표 3. 매크로 블록 각 위치에서의 클록수
Table 3. Clocks at each position of MB.

Y	X	clk/1MB	MB(s)	total clks
0	0	5332	1	5,332
0	GEN	6813	9	61,317
0	MAX	6474	1	6,474
not 0	0	6857	8	54,846
not 0	GEN	9443	72	679,176
not 0	MAX	9059	8	72,472
clks / 1 frame			99	879,617

* GEN = not 0 && not MAX

표 4. 각 모듈의 게이트 수
Table 4. Gate counts of each module.

MODULE	SIZE
Controller	4.48 K
Prediction Calculator	1.78 K
PLANE Calculator	2.32 K
ETC.	3.3 K
TOTAL	11.9 K

표 5. 성능 비교
Table 5. Performance comparison.

	본 논문	[3]	[8]
구조 특징	sequential	4 parallel	4 parallel
	SAD	SATD	SATD
면적 [gates]	약 11.9K	약 28K	약 36K
메모리 크기[byte]	1078	< 2048	< 2048
공정	Samsung 0.18um	TSMC 0.25um	UMC 0.18um
최대 동작 속도	106.8MHz	55MHz	125MHz
clks /MB	약 8,885	< 1,300	< 1,080
clks / 1 QCIF frame	879,617	< 128,700	< 106.920

4.8%의 로직 리소스와 약 8.6Kbit의 메모리를 사용하였다. 클록은 그림 13.(a)에서 보이는 바와 같이 왼쪽이나 위쪽 가장자리가 아닌 경우 모든 모드를 수행하는데 걸리는 최악의 경우는 9,443 클록이다. 그 외의 경우까지 고려한 1 프레임당 소요되는 인트라 예측기의 실행시간은 표 3과 같다.

구현된 인트라 예측기는 Samsung STD130 0.18um CMOS Standard Cell Library와 Synopsys사의 Design Analyzer를 이용하여 합성하고 검증하였다. 그 결과, 최장 지연 경로는 9.36ns이며 인터페이스를 제외한 로직 리소스는 11.9K 게이트가 사용되며 1,078Kbyte의 SRAM 메모리가 쓰였다. 각 모듈 당 필요한 로직 게이트수가 표 4에 나타나 있다.

구현한 인트라 예측기의 처리 속도를 살펴보게 되면 QCIF(176픽셀 x 144픽셀) 크기의 영상을 기준으로 75MHz의 동작 속도를 가진다고 했을 때 초당 85.26 프레임 처리가 가능하다. 이는 실시간으로 처리 가능한 속도이다.

기존 인트라 예측기와 본 논문에서 제안한 구조를 표 5에서 비교해 보았다. 1개의 매크로블록을 처리하는데 소요되는 클록, 칩으로 구현하였을 때 하드웨어 구조에서의 메모리 크기 그리고 구현된 구조의 크기를 비교하였다. 기존의 구조는 4개의 연산기를 두어 4개의 픽셀을 동시에 처리하고 모드 결정을 위해 차분 영상에 하다마드 연산을 수행하는 SATD를 채택한 반면 본 구조에서는 하나의 연산기만을 둔 구조를 채택하였다. 그리고 하다마드 연산을 수행하지 않고 차분값만으로 모드 결정을 하는 SAD연산으로 계산량을 줄여서 모드 결정을 수행하였다. 그 결과 [3]의 42.5% [8]의 33.1% 정도의 적은 게이트 수를 사용하는 결과를 얻었다. 이는 모바일 환경에서 기존의 논문들보다 우수한 성능을 나타낼 수 있음을 보여준다.

V. 결론 및 향후 계획

본 논문에서는 H.264/AVC를 위한 인트라 예측의 알고리즘을 기술하고 모바일 환경을 위한 하드웨어 구조를 제안하였다. 적은 게이트수와 메모리 크기를 작게 하기 위해 하나의 연산기와 SAD 연산을 통한 인트라 예측기를 사용하였고, 참조 메모리를 최적화시키기 위해 전 픽셀의 부분만을 저장 하는 방법을 사용하였다., 구현된 모듈들을 ASIC 라이브러리와 합성한 결과를 바탕으로 성능을 분석하면 QCIF(176x144) 와 CIF (352x288) 영상에 대해서 각각 초당 121.5와 30.36의 프레임 처리가 가능하다. 이는 H.264/AVC Baseline 영상 시스템에 적용 가능한 성능임을 보여준다. 현재는 전체 H.264/AVC I-Frame 부호화기의 통합 최적화 과정에 있으며, 향후 인터 추정기의 성능을 최적화 시켜 H.264/AVC 전체 부호화기를 설계할 예정이다.

참 고 문 헌

[1] 호요성, 김승환, "H.264/AVC 표준의 소스 코드 분석", 두양사, 83쪽, 2006.

[2] 大久保. 榮, 角野 眞也, 菊池 義浩, 鈴木 輝彦, "H.264/AVC 教科書", 홍릉과학출판사, 124-131쪽, 229-236쪽, 2006.

[3] Y. W. Huang, B. Y. Hsieh, T. C. Chen, L. G. Chen, "Analysis, Fast Algorithm, and VLSI Architecture Design for H.264/AVC Intra Frame Coder," Circuits and Systems for Video Technology, IEEE Transactions on , Volume: 15 , Issue: 3 , pp. 378 - 401, March 2005.

[4] Iain E.G Richardson "H.264 and MPEG-4 Video Compression", WILEY, pp.180-183, 2003.

[5] Y. Lim, D. Lee and Y. Jeong, "Hardware Implementation of a Fast Inter Prediction Engine for MPEG-4 AVC", Journal of the Korea Institute of Communication Sciences(KICS), 2005.

[6] Joint Video Team Reference Software JM8.6 [Online].
Available: http://bs.hhi.de/suehring/tml/download/

[7] ARM , AMBA Specification (Rev 2.0), ARM IHI 0011A, 1999.

[8] C. C. Cheng, C. W. Ku, and T. S. Chang, "A 1280x720 pixels 30 frames/s H.264/MPEG-4 AVC intra encoder," Proc. IEEE International Symposium on Circuits and Systems, May. 2006.

저 자 소 개

윤 철 환(학생회원)
2006년 2월 서경대학교 컴퓨터 공학과 학사 졸업.
2006년 3월~현재 광운대학교 전자통신공학과 석사과정
<주관심분야 : 고성능 압축코덱, 영상처리, 임베디드 시스템, 컴퓨터 연산 알고리즘, ASIC 설계>

정 용 진(정회원)
1983년 서울대학교 제어계측 공학과 학사 졸업.
1983년 3월~1989년 8월 한국전자통신연구원
1995년 2월 미국 UMASS 전자전산공학과 박사
1995년 4월~1999년 2월 삼성 전자 반도체 수석 연구원
1999년 3월 광운대학교 전자공학부 부교수
<주관심분야 : 무선 통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템>