Journal of Korea Multimedia Society Vol. 12, No. 2, February 2009(pp. 200-208)

JBIG2 허프만 부호화기의 하드웨어 설계

박경준[†], 고형화[††]

요 약

JBIG2는 차세대 이진 영상의 압축 표준으로서 차세대 팩스가 실용화되기 위해서는 임베디드 장비에서 사용가능한 하드웨어 모듈의 제작이 필수적이다. 본 논문에서는 JBIG2의 부호화에서 핵심이 되는 고속 허프만 부호화기의 하드웨어 모듈을 제안하였다. 모두 15개의 허프만 부호화 테이블을 메모리에 저장하여 선택적으로 이용하도록 하였다. 본 논문은 최소의 필요 데이터만을 이용하여 허프만 부호화를 하도록 설계하여 효율적으로 메모리를 사용함으로써 고속의 처리가 가능하도록 제안하였다. 설계된 허프만 부호화기는 Xilinx의 Virtex-4 FPGA칩에 포팅하여 임베디드 보드상에서 Microblaze 코어를 이용한 소프트웨어 모듈의 연동 실행이 가능하도록 구현하였다. 설계된 허프만 IP 모듈은 시뮬레이션과 연동 실험 및 검증을 통하여 성공적으로 동작함을 확인하였다. 효율적 메모리 이용에 의한 하드웨어 설계로 임베디드 시스템 상에서 소프트웨어만으로 실행한 것 보다 10배 이상의 빠른 처리 속도를 나타내었다.

Hardware Design for JBIG2 Huffman Coder

Kyung-Jun Park[†], Hyung-Hwa Ko[††]

ABSTRACT

JBIG2, as the next generation standard for binary image compression, must be designed in hardware modules for the JBIG2 FAX to be implemented in an embedded equipment. This paper proposes a hardware module of the high-speed Huffman coder for JBIG2. The Huffman coder of JBIG2 uses selectively 15 Huffman tables. As the Huffman coder is designed to use minimal data and have an efficient memory usage, high speed processing is possible. The designed Huffman coder is ported to Virtex-4 FPGA and co-operating with a software modules on the embedded development board using Microblaze core. The designed IP was successfully verified using the simulation function test and hardware-software co-operating test. Experimental results shows the processing time is 10 times faster than that of software only on embedded system, because of hardware design using an efficient memory usage.

Key words: JBIG2, Huffman(허프만), SoC(시스템온칩), IP(Intellectual Property)

1. 서 론

JBIG2(Joint Bi-level Image Experts Group) 는 차세대 이진영상의 압축 표준으로 제정된 국제 표준[1-4]으로서 팩스나 이진 문서 압축 등에 이용될 수 있다. JBIG2가 이진 문서의 압축을 위해 표준화된 후 JPEG, MPEG 등의 다른 표준화 등에 비해 활발한 응용이 이루어지고 있지 않다. 현재 사용되고 있는 팩스 규격인 G3방식의 출력물의 품질상태가 매우 낮다. 반면에, JBIG2의 이진 문서에 대한 압축 성능은 G4방식보다 3~5배, JBIG방식보다 2~4배 정도 우수하다[5,6]. 따라서, 현재 G3 및 G4 팩스보다 동일한

※ 교신저자(Corresponding Author) : 박경준, 주소 : 서울시 노원구 월계동 447-1(139-701), 전화 : 02)940-5137, FAX : 02)940-5137, E-mail : red2000@kw.ac.kr
접수일 : 2008년 11월 12일, 완료일 : 2009년 2월 2일
[†] 정회원, 광운대학교 전자통신공학과 박사과정
[††] 정회원, 광운대학교 전자통신공학과 교수
(E-mail : hhkoh@kw.ac.kr)
※ "이 논문은 2007년도 광운대학교 연구년에 의하여 연구되었음."

속도로 훨씬 우수한 품질의 문서를 전송할 수 있다. 또, 여러 페이지의 문서를 압축하여 전송할 경우에는 압축 성능이 더 높아진다. 차세대 팩스 시스템이 필요한 이유이기도 하다. 또한 기존의 종이 문서의 디지털화에 있어서도 그 활용 가능성은 높이 평가될 수 있다.

JBIG2 표준화의 가장 많이 활용될 것으로 예측되어 온 차세대 팩스 시스템은 전용 하드웨어의 개발이 요구되지만 이에 대한 연구가 미흡한 상태이다. 현재 JBIG2 압축 알고리즘은 전자문서 형식으로 널리 사용되고 있는 PDF 형태의 파일에서 채택하고 있다. Adobe사의 Acrobat[7]에서 문서영상의 압축 알고리즘으로 사용되고 있으며, Lead Technology사의 LEADTOOS 프로그램[8], Verypdf사의 Image2PDF 프로그램[9] 등 PDF 파일을 지원하는 프로그램들에서 JBIG2 압축 알고리즘을 채택하여 사용하고 있다. 그러나 현재 소프트웨어로만으로 이용되고 있는 JBIG2는 컴퓨터상에서만 사용할 수 있다는 한계가 있다. JBIG2 표준의 응용분야로 유력한 팩시밀리 분야의 경우에는 컴퓨터상에서 운영체제에 종속되어 동작되기보다 임베디드 시스템에서 사용되도록 구성해야 하므로 C언어 등으로 구현된 소프트웨어 처리 방식보다는 VHDL 등의 하드웨어로 개발하는 것이 필요하다. 이러한 JBIG2 부호기를 하드웨어로 설계 제작하면 차세대 팩시밀리의 개발을 앞당기고, 고속의 처리가 가능하므로 산업적인 측면에서 의의가 있다. 현재 이러한 연구는 제한적으로 이루어지고 있고 결과가 발표된 사례를 찾아보기 힘들다.

그러므로, 임베디드 시스템에서 동작할 수 있는 JBIG2 압축시스템을 구성하고, 이에 관련된 핵심 모듈의 SoC/IP(System on Chip/Intellectual Property)의 개발은 중요한 과제이다. JBIG2 부호기는 ARM 또는 PPC 등의 임베디드용 프로세서를 이용하여 소프트웨어만으로 동작할 경우를 하드웨어로 제작과 비교해 보면 처리속도가 늦고 메모리의 부족으로 정상적인 동작 수행에는 많은 어려움이 있다. 이를 극복하기 위하여서는 SoC/IP 형태로의 설계 제작이 필요하다. JBIG2 부호기 전체를 하드웨어로 개발이 필요하지만 개발과정에서 일부 기능은 하드웨어로 설계 제작하고 기타 기능은 소프트웨어로 연동되는 연동 모델의 개발이 동시에 필요하다. 본 논문에서는 허프만 부호화기 하드웨어 IP의 효율적인 설계와 기존의 소프트웨어와의 연동 가능한 모델을 제안하였다.

본 논문의 구성은 2장에서 JBIG2의 전체적인 구조에 대하여 설명하였고, 3장에서는 허프만 부호기의 구조와 이를 하드웨어 IP로 설계한 방법에 대하여 설명하였고, 4장에서는 설계된 하드웨어의 검증을 통하여 하드웨어 IP로 정상 동작함과 우수한 처리 성능을 확인하였고, 5장에서 결론을 이끌어 내었다.

2. JBIG2 소개

JBIG2 표준은 이진 영상 압축 알고리즘의 표준으로 팩스나 문서의 전송에 이용되고 있는 기존의 MH (Modified Huffman), MR(Modified Read), MMR (Modified Modified Read), JBIG 등의 알고리즘이 발전된 형태로 SPM(Soft Pattern Matching)알고리즘을 기반으로 한다. JBIG2 표준은 비트 스트림의 구성과 복호기에 관한 것만을 표준화로 정의하고 있으며, 부호기는 표준화되어 있지 않다. 표준안에서의 전체 복호기 블록은 문서(Text) 영역, 일반(Generic) 영역과 패턴(Pattern) 영역의 복호화로 구분된다[1].

일반적으로 JBIG2 알고리즘은 이진 영상으로부터 심벌을 찾아내고, 심벌의 집합인 심벌사전(Symbol Dictionary)을 만들어서 그 정보들을 이용하여 문서를 압축하는 방법이다. 입력된 이진 영상으로부터 찾아낸 심벌이 심벌 사전 내에 존재할 경우 심벌의 색인(Index)과 위치정보만을 이용하고, 심벌 사전에 존재하지 않을 경우에는 이를 심벌 사전에 추가하는 방식으로 압축이 수행된다. 이러한 압축 방법은 여러 페이지의 문서를 압축할 경우 효과적인 압축 성능을 보인다. JBIG2 표준은 손실 모드와 무손실 모드를 모두 제공하고 있으므로 사용자의 환경 및 효용성에 따라 선택할 수 있도록 하였다.

JBIG2 부호화 과정 중 제일 먼저 수행되는 작업은 심벌을 찾는 것이다. 심벌은 하나로 연결된 객체를 의미한다. 심벌을 찾아내는 방법은 윤곽선 추출법을 사용한다. 찾아진 심벌들은 심벌 사전에 동일한 심벌이 있는지를 알아보고, 없으면 새로운 심벌로 판별되어 심벌 사전에 등록된다. 이렇게 등록된 심벌들은 먼저 심벌의 높이(Height) 순으로 정렬(Sort)되어 같은 높이를 가지는 심벌들끼리 묶어 클래스(Class)를 만든다. 가장 작은 높이를 가지는 심

벌들이 첫 번째 높이 클래스(First Height Class)가 되고, 그 다음이 두 번째 높이 클래스(Second Height Class) 순으로 정렬되며, 각 클래스 간의 높이 차이가 전송된다. 각각의 클래스 안에서의 심벌들은 폭(Width) 순서대로 정렬하게 된다. 폭이 가장 작은 값부터 순서대로 정렬되어 앞 심벌과의 폭의 차이를 전송한다. 이런 값들은 허프만 부호화를 이용한다. 그리고 심벌의 비트맵은 허프만 부호화를 이용할 경우는 MMR 부호화를 위해서 같은 높이를 가진 심벌들이 결합되어 하나의 비트맵을 이룬 후 부호화 되고, 허프만 부호화를 이용하지 않는 경우의 비트맵은 각각 폭의 차 값을 부호화 한 후에 각각의 비트맵을 부호화 하게 된다.

텍스트 영역 부호화는 심벌사전에 등록된 심벌들과 심벌영역에서의 심벌 색인과 심벌의 S위치(가로)와 T위치(세로)를 이용하여 심벌이 놓이게 될 위치를 결정하게 된다. 여기서 심벌은 스트립(Strip)이라는 단위로 나뉘어 2진 문서를 높이에 따라 분할하게 된다. 이 스트립은 문서의 분할된 높이를 결정하는데 비트맵 라인 1, 2, 4, 8 중 하나를 선택한다. 각각 그룹화된 스트립 단위 내에서 심벌들은 이전 심벌들과의 S위치와 T위치의 차이와, 색인정보만을 전송한다. 첫 스트립에서 첫 번째 심벌의 DFS(Delta First S)값은 심벌영역의 시작 위치 S를 기준으로 차를 부호화하고 첫 스트립이 아닌 경우에는 이전 스트립의 첫 심벌의 위치를 기준으로 차 값을 부호화 한다.

그 밖의 일반 영역의 부호화는 1과 0의 직사각형의 비트맵을 직접 부호화 방식을 의미한다. 차례대로

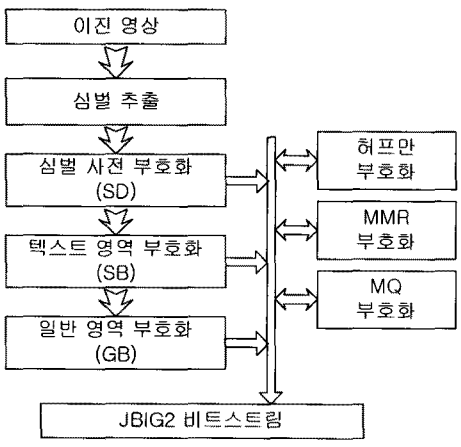


그림 1. JBIG2 부호화기의 전체 블록도

각 화소의 값을 부호화하며, 각 화소를 효율적으로 부호화하기 위하여 주변의 화소를 이용한다. MMR 부호화를 이용할 경우에는 이전 라인과 현재 라인의 상대적인 위치를 이용하여 효율적인 부호화를 수행한다. 산술부호화를 이용할 경우에는 각 화소가 마르코프 정보원이란 가정에서 각 화소의 값을 예측하는 템플릿을 이용하여 부호화를 수행한다. 마르코프 정보원이란 임의의 화소가 0 또는 1을 취할 확률이 그 이전에 발생한 m개의 화소 참조에 의하여 결정된다고 할 때 m+1개의 이상으로 참조 화소의 수를 증가시켜도 부호화 화소에 대한 조건부 확률이 바뀌지 않는 것을 의미한다. 참조 화소가 취하는 상태를 콘텍스트(Context)라 하며, JBIG2에서는 네 가지 형태의 화소 배치를 가지는 템플릿을 정의하고 있다.

3. 허프만 부호화기의 하드웨어 설계

3.1 JBIG2 허프만 부호화기

JBIG2 부호화기는 허프만(Huffman) 부호화와 MQ 산술부호화를 선택적으로 사용할 수 있어서 압축은 낮지만 빠른 수행을 원할 경우에는 허프만 부호화를 사용하고, 높은 압축 성능을 요구할 경우에는 MQ 산술부호화기를 선택하여 사용한다.

본 논문에서는 고속의 처리를 위한 허프만 부호화기를 하드웨어로 개발하였다. 설계하고자 하는 허프만 부호화기는 심벌 사전이 만들어지면, 이를 부호화하는데 사용되어지며, 또 심벌사전의 데이터를 이용한 심벌의 위치를 결정하는 텍스트 영역(Text Region) 부호화에도 이용되어진다.

허프만 부호기는 가변길이 부호로서 데이터의 발생 빈도수에 따라 부호를 할당함으로써 효율적인 사용이 가능하도록 하는 데이터 압축 기술로서 데이터의 발생 빈도수에 따른 허프만 부호를 생성해야 한다. 이는 모든 데이터를 이용하여 발생 확률을 확인해야 하므로 데이터 처리에 많은 시간과 노력이 소요된다. 이를 고속의 처리가 가능하도록 하기위하여 미리 만들어진 허프만 테이블을 이용한다. JBIG2 에서는 모두 15개의 허프만 테이블을 이용하여 빠른 부호화 처리를 수행하도록 정하고 있다[1]. 각 허프만 테이블은 필요에 따라 선택적으로 사용하도록 되어있다. 허프만 테이블의 내용에는 HTOOB라는 허프만 테이블의 OOB(Out-Of-Band) 를 정의하고 있으며,

부호화 대상 구간과 해당하는 구간의 프리픽스 길이(Prefix Length)와 구간 길이(Range Length), 프리픽스 부호(Prefix Code)와 구간 부호(Range Code)가 정의되어 있다. 부호화하고자 하는 데이터 값은 해당하는 범위의 구간 안에 존재할 경우 프리픽스 길이와 구간 길이에 의해서 허프만의 가변길이 부호가 정해진다. 표 1과 표 2는 JBIG2 표준안에서 제시한 허프만 부호기의 15개의 테이블 중 2번과 6번, 2개의 테이블을 보여준 것이다.

허프만 부호화는 심벌사전 부호화와 심벌의 위치를 결정하는 텍스트 영역 부호화에 사용된다. 각 부호화를 수행할 데이터에 따라 사용되어지는 허프만 테이블이 결정된다. 심벌 사전의 경우에서는 비트맵 크기의 정보는 테이블 1번을, 심벌들 간의 폭의 차이(Delta Width)는 테이블 2번 또는 3번 중 하나를, 심벌들 간의 높이 차이(Delta Height)이는 테이블 4번 또는 5번 중 하나를 선택하여 허프만 부호화를 수행한다. 텍스트 영역을 허프만 부호화할 경우에서는 첫 심벌의 X좌표(폭)는 테이블 6번이나 7번 중 하나를, 심벌 간에 X좌표(폭)의 차이는 테이블 8번, 9번, 10번 중 하나를, 심벌 간에 Y좌표(높이)의 차이는 테이블 11번, 12번, 13번 중 하나를 선택한다. 또, 각 심벌들 간에 폭의 차이, 높이의 차이, X좌표의 차이, Y좌표의 차이를 정련(Refinement)할 때는 테이블 14번 또는 15번 중 하나를 선택하여 허프만 부호화를 수행한다.

3.2 허프만 부호화기의 하드웨어 IP 설계

허프만 부호화기의 하드웨어 IP설계는 JBIG2 전체 시스템 중 일부로 실제 JBIG2 비트스트림을 생성하는데 큰 역할을 하는 핵심 중 하나이다. 빠른 가변길이 부호화를 하는데 효율적이어서 JBIG2 부호화 수행의 기본이라 할 수 있다. JBIG2 에서의 허프만 부호화기는 선택한 허프만 테이블과 부호화 하려는 값에 의해 부호화가 진행된다.

허프만 부호화기의 하드웨어 IP는 기존의 JBIG2

표 1. 표준 허프만 테이블 2번

HTOOB	1		
값 (VAL)	프리픽스길이(PREFLEN)	구간길이(RANGELEN)	부호화 (Encoding)
0	1	0	**0**
1	2	0	**10**
2	3	0	**110**
3...10	4	3	**1110**+(VAL-3) encoded as 3 bits
11...74	5	6	**11110**+(VAL-11) encoded as 6 bits
75...∞	6	32	**111110**+(VAL-75) encoded as 32 bits
OOB	6		**111111**

표 2. 표준 허프만 테이블 6번

HTOOB	0		
값 (VAL)	프리픽스길이	구간길이	부호화 (Encoding)
-2048...-1025	5	10	**11100**+(VAL+2048) encoded as 10 bits
-1024...-513	4	9	**1000**+(VAL+1024) encoded as 9 bits
-512...-257	4	8	**1001**+(VAL+512) encoded as 8 bits
-256...-129	4	7	**1010**+(VAL+256) encoded as 7 bits
-128...-65	5	6	**11101**+(VAL+128) encoded as 6 bits
-64...-33	5	5	**11110**+(VAL+64) encoded as 5 bits
-32...-1	4	5	**1011**+(VAL+32) encoded as 5 bits
0...127	2	7	**00**+VAL encoded as 7 bits
128...255	3	7	**010**+(VAL-128) encoded as 7 bits
256...511	3	8	**011**+(VAL-256) encoded as 8 bits
512...1023	4	9	**1100**+(VAL-512) encoded as 9 bits
1024...2047	4	10	**1101**+(VAL-1024) encoded as 10 bits
-∞...-2049	6	32	**111110**+(-2049-VAL) encoded as 32 bits
2048...∞	6	32	**111111**+(VAL-2048) encoded as 32 bits

소프트웨어 프로그램과의 연동이 가능하도록 설계하였다. 그림 2는 제안한 허프만 부호화기의 하드웨어 IP의 내부 구조를 보여주고 있다. 허프만 부호화기의 하드웨어의 입력으로는 32비트의 데이터 스트림을 두 번 연속으로 전송받도록 설계하였다. 전송되는 데이터로는 OOB값과 사용되는 허프만 테이블 번호를 16비트씩 연속하여 전송하고, 다음 32비트로는 부호화할 데이터를 전송하도록 설계하였다. 입력된 데이터를 이용하여 허프만 부호화테이블의 값을 미리 저장된 메모리로부터 얻어서 부호화할 데이터 값에 해당하는 부호를 찾아내어 허프만 부호를 전송하도록 하였다. 허프만 부호화에 사용될 15개의 허프만 부호화 테이블들은 부호화할 데이터의 구간별 최저값, 프리픽스 길이와 프리픽스 부호, 구간 길이가 별도의 메모리에 저장되어 있다.

허프만부호화기의 동작은 먼저 선택된 허프만 테이블의 정의된 값을 메모리로부터 읽어서 기본적인 환경 값을 설정한다. 다음으로는 해당되는 허프만 테이블의 구간별 최저값을 메모리로부터 읽어서 부호화될 데이터의 값과 비교하여 어느 구간에 속하여 있는지를 찾아내어 그 위치에 해당하는 색인을 이용하여 프리픽스 길이와 프리픽스 부호, 그리고 구간 길이를 메모리로부터 읽어온다. 구간 부호는 부호화될 값에서 해당구간의 최저값과의 차이가 부호로 만들어진다.

허프만 부호화기는 구간별 최저값의 경우 16비트 데이터로, 프리픽스 길이, 구간 길이의 경우는 8비트 데이터로, 프리픽스 부호는 16비트 데이터 형식으로 메모리에 저장되어 있다. 허프만 테이블의 경우 프리픽스 Code의 경우 최대 9비트를 구간 Code의 경우 최대 32비트를 사용하게 된다. 따라서 허프만 부호화된 데이터 값의 출력은 32비트의 스트림을 2회 출력함으로서 부호화된 데이터를 모두 전송하게 된다. 출력에 이용되는 데이터는 프리픽스 길이 8비트(31:24), 구간 길이 8비트(23:16), 프리픽스 부호로 16비트(15:0)의 데이터 값이 할당이 되고, 구간 부호로 32비트(31:0)의 데이터 값이 할당이 된다.

그림 3은 설계된 허프만 부호화기의 동작의 순서를 나타낸 상태도를 나타낸 그림이다.

S0 : 하드웨어 초기화를 수행한다.

S1 : 2번의 연속된 32비트의 데이터를 입력받는다. (OOB, 허프만 테이블 번호, 데이터)

S2 : OOB가 0일 경우 해당 허프만 테이블의 구간별 최저값을 설정한다.

S3 : 입력된 데이터 값이 구간의 최저값보다 작을 경우 메모리로부터 해당 부호를 읽는다.

S4 : 입력된 데이터 값이 구간의 최고값보다 클 경우 메모리로부터 해당 부호를 읽는다.

S5 : 입력된 데이터 값이 속한 구간을 찾는다. 해당 구간을 찾을 때까지 반복 수행한다.
해당구간을 찾으면 메모리로부터 해당 부호를 읽는다.

S6 : HTOOB가 1일 경우 OOB에 해당하는 부호를 메모리로부터 읽는다.

S7 : 해당 부호를 2번의 연속된 32비트의 스트림으로 출력한다. (Prefix Length, Prefix Code, Range Length, Range Code)

S8 : 하드웨어 동작을 종료한다.

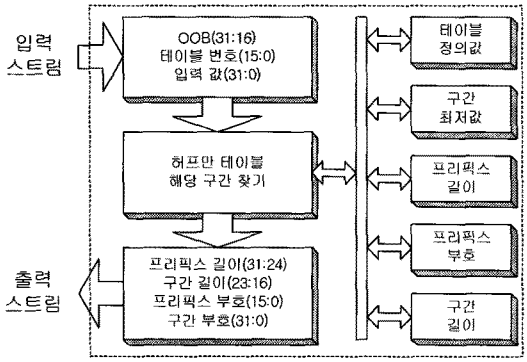


그림 2. 설계된 허프만 부호화기 하드웨어 내부 구조

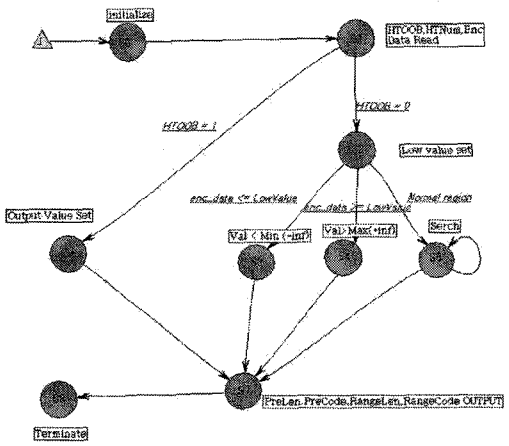


그림 3. 설계된 허프만 부호화기 하드웨어 상태도

본 연구를 통해 설계된 허프만 부호화기의 하드웨어 구조의 특징은 허프만 테이블의 모든 데이터 값을 메모리로부터 읽어올 필요가 없으며, S2 상태에서 해당 테이블의 구간별 최저값만을 먼저 읽어서 비교과정을 수행함으로써 메모리로부터 읽어오는 데이터를 최소화 하였다. S7 상태에서 부호화 결과 데이터도 해당 구간의 값만을 다시 메모리로부터 읽어서 출력하게 함으로써 허프만 부호화의 효율적인 메모리 구성과 처리시간을 최소화 하였다. OOB의 경우에는 비교수행을 하지 않아 해당 테이블의 구간별 최소 값을 읽어올 필요가 없으므로 수행시간을 단축시킬 수 있다. 허프만 부호화를 수행할 때마다 모든 허프만 부호화 테이블의 정보를 메모리로부터 읽어온다면 허프만 부호화 수행에 있어서 메모리 접근에 대부분의 시간이 소요될 것이다. 그러나 이 메모리 접근을 최소화한다면 더 빠른 시간 내에 처리가 가능할 것이다. 이를 위하여 허프만 부호화에 필요한 최소의 데이터를 먼저 읽어오고, 이를 이용하여 허프만 부호화가 수행될 테이블의 해당 구간을 찾아 해당 데이터만을 다시 메모리로 읽어 오도록 설계함으로써 고속의 처리가 가능하도록 설계하였다.

허프만 부호기 하드웨어 IP의 동작을 검증하기 위하여 허프만 부호화기의 소프트웨어와 하드웨어를 연동되도록 실험하였다. 그림 4에 소프트웨어와 하드웨어 연동시스템의 블록도를 보였다. 실제로 허프만 부호화가 수행되어지는 곳은 Do_Huff 부분으로 이곳에서 데이터를 입력받아 허프만 부호화 값을 만들어 내는 곳이다. 다른 부분은 소프트웨어와 하드웨어의 데이터 전송을 위하여 설정하는 부분으로 제어신호인 데이터 전송을 알리는 rdy(ready)신호, 전송포트의 사용을 위한 en(enable) 신호, 데이터 전송

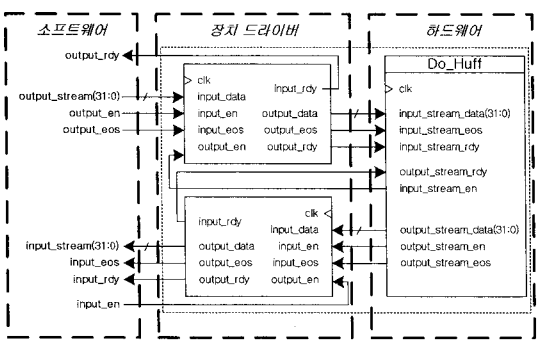


그림 4. 허프만 부호화기의 하드웨어와 소프트웨어의 연동 구조

종료를 알리는 eos(end of stream) 신호와 데이터가 전송이 이루어지는 32비트의 스트림(stream)으로 구성되어진 장치 드라이버(device driver)를 설명한 것이다. 연동을 위하여 소프트웨어와 하드웨어간의 데이터 전송에 있어서 먼저 전송포트를 연결하는 en신호를 보내고, 데이터의 전송을 시작하기 위하여 rdy 신호를 보낸다. 32비트의 데이터 전송이 이루어지고, 데이터 전송이 끝나면 eos신호를 보냄으로 전송이 종료되었음을 알린다.

설계된 하드웨어 IP는 VHDL 언어로 작성되었고, Impulse사의 CoDeveloper를 이용하여 시뮬레이션을 수행하여 정상적으로 동작함을 확인하였다[10].

4. 실험 및 결과

4.1 실험 방법 및 설계 검증

설계된 허프만 부호화기 IP는 Xilinx 사의 ISE 와 EDK Tool을 이용하여 하드웨어로 합성하였다. 실험에 이용된 임베디드 보드는 그림 5와 같이 Xilinx사의 Virtex-4 FX60 FPGA 칩을 사용하고 있는 ML410 보드이다. FPGA 내부에 Microblaze Core를 구현하여 100MHz의 동작속도로 실험 및 검증을 수행하였다[11].

표 3은 설계된 허프만 부호화기의 Virtex-4 FX60 FPGA에서의 사용량을 보여주고 있다. Xilinx ISE를 이용하여 FPGA 사용량을 측정하였는데 639 slices를 사용하여 전체 사용가능한 용량의 2% 정도를 사용하였음을 보여주고 있다.

기능 검증은 Active-HDL 프로그램을 이용하여 파형 시뮬레이션 테스트를 통해 수행하였다. 허프만 부호화기에 테스트 신호를 입력하여, 그 출력 결과를 확인함으로써 정상적인 동작함을 알 수 있었다. 그림 6에 실험 결과를 보여준다. 동작 주파수 100MHz로

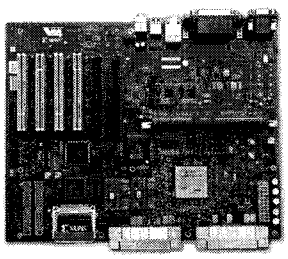


그림 5. 실험에 사용한 ML410 테스트 보드

표 3. 허프만 부호화기의 FPGA 사용량

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utiliz -ation**
Num. of Slice Flip Flops	371	50,560	1%
Num. of 4 input LUTs	966	50,560	1%
Logic Distribution			
Num. of occupied Slices	639	25,280	2%
Total Num. 4 input LUTs	1,100	50,560	2%
Num. used as logic	966		
Num. used as a route-thru	2		
Num. used for Dual Port RAMs	132		
Num. of bonded IOBs	72	576	12%
Num. of BUFG/ BUFGCTRLs	1	32	3%
Num. of FIFO16/ RAMB16s	6	232	2%
Total equivalent gate count for design	411,652		

동작하였을 경우 32비트의 데이터가 두 번 입력이 되면 출력 데이터가 32비트로 두 번 출력되어 허프만 부호화기의 하드웨어 모듈의 동작시간이 약 605ns가 소요됨을 알 수 있었다. 이 동작 시간은 허프만 하드웨어 모듈이 한 번 수행된 동작 시간을 의미한다. 허프만 테이블 번호 2번을 선택하여 25란 값을 입력하여, 프리픽스 길이가 5이고, 프리픽스 부호가 30, 구간 길이가 6으로 구간 부호가 14의 출력을 가지므로 앞의 표 1의 허프만 테이블을 참고하여 정상동작하고 있음을 알 수 있다.

4.2 성능 평가

설계된 허프만 부호화기는 ML410 보드의 FPGA에 Xilinx EDK tool 을 이용하여 하드웨어 IP로 추가하였고, 하드웨어 동작 실험을 통하여 32비트의 연속된 데이터를 하드웨어로 전달하고 허프만 부호화된 데이터를 전송받아 그 결과를 확인하는 방식으로 수행 검증과 성능평가를 수행하였다. 결과의 확인은 UART 터미널을 이용하여 확인하였다. 그림 7은 설계된 허프만 부호화기의 하드웨어 동작 테스트 결과를 보여주고 있다. 허프만 부호화에 이용될 데이터를 하드웨어로 전송하면, FPGA IP로 구성된 하드웨어에서 허프만 부호화가 수행되어진 결과를 전송받아 그 결과를 터미널 창에 표시하도록 한 것이다. 터미널 창에는 선택된 허프만 테이블 번호와, OOB여부, 부호화될 데이터와 허프만 부호화의 결과인 프리픽스 길이와 부호, 구간 길이와 부호를 여러 번 수행하여 표시하도록 하고, 표시된 결과를 확인함으로써 모두 정상적으로 동작함을 알 수 있었다.

설계된 허프만 부호화기의 성능 평가를 위하여 허프만 부호화를 Xilinx의 Microblaze 코어를 이용하여 임베디드 보드상에서 소프트웨어로 수행하였을 경우와 하드웨어 IP로 수행하였을 때의 수행시간을 비교 측정하였다. 수행평가는 설계환경이 Microblaze를 이용하였기 때문에 Xilinx FPGA의 OPB_Timer를 이용하여 수행 시간을 비교함으로서 평가 하였다.

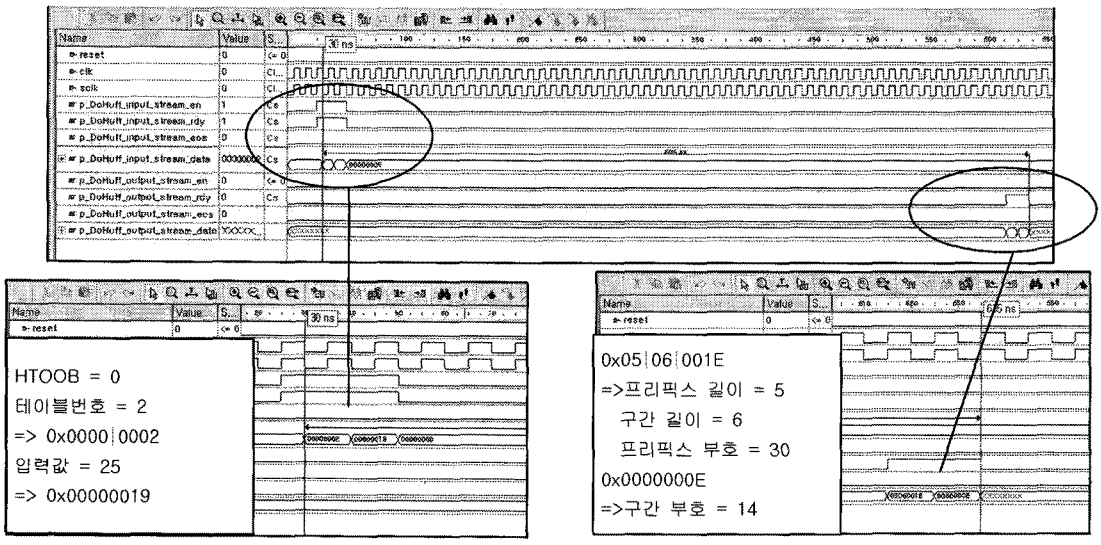


그림 6. 허프만 부호화기 하드웨어 시뮬레이션 결과

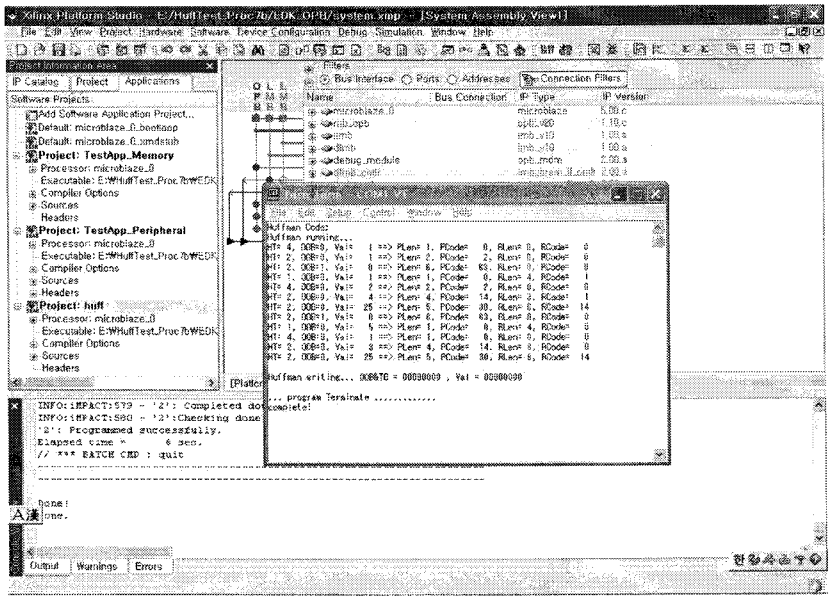
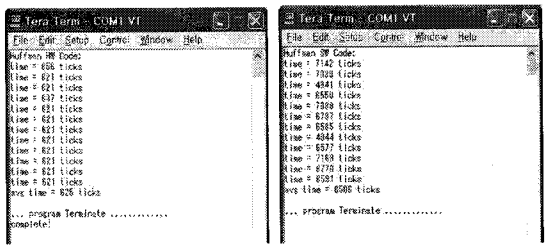


그림 7. 허프만 부호화기 하드웨어의 동작 테스트 결과

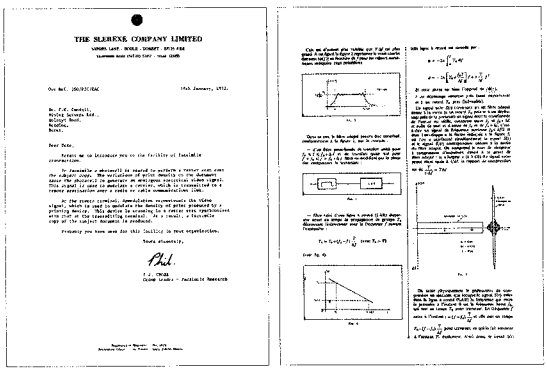
그림 8은 허프만 부호화의 수행 속도 비교 실험 결과를 캡쳐한 것이다. Xilinx ML410 보드에서 클럭 주파수가 100MHz에서 동작하는 실험 환경에서 한 번의 허프만 부호화를 수행하였을 경우 소프트웨어로만 동작하였을 경우 평균 6505ticks 의 시간이 소요되었고, 하드웨어를 이용하였을 경우 평균 625ticks 의 시간이 소요되었다. 여기서 ticks라 함은 수행된 시간동안의 사용된 클럭수를 의미한다. 다시 말해 수행 결과 ticks수에 클럭을 곱하면 수행시간을 의미한다. 625ticks는 클럭의 주기가 10ns에 해당하므로 6250ns의 수행 시간을 의미한다. 실험은 12번의 허프만 동작을 연속 수행한 결과의 평균값을 나타낸 것이다. 이러한 결과로써 하드웨어로의 동작이 소프트웨어로의 동작 속도보다 약 10배 이상 빠른 수행을 보여 하드웨어 설계의 필요성을 보여주고 추후 JBIG2 ASIC 칩의 설계에 활용이 기대된다.



(a) (b)

그림 8. 허프만 부호화기의 속도 비교. (a) 하드웨어와 소프트웨어의 연동 수행, (b) 임베디드 소프트웨어만으로 수행

설계된 허프만 부호화기 하드웨어 모듈은 JBIG2 소프트웨어 모듈과의 연동 테스트를 통해 이진 문서의 압축을 수행하였다. 실험에 사용한 영상은 JBIG 위원회에서 공식적으로 사용하는 테스트 문서 중에서 200dpi(dot per inch) 해상도를 가진 CCITT-1(텍스트 위주)과 CCITT-5(그림이 포함)문서이다. 연동 실험은 이진 문서 영상파일을 허프만 부호화를 이용한 JBIG2 압축을 수행하여 파일(.jb2)을 생성하였고,



(a) (b)

그림 9. 연동실험 결과 복원된 이진 문서영상. (a) CCITT-1 200dpi, (b) CCITT-5 200dpi

복원을 수행하여 만들어진 이진 문서영상을 확인 해 본 결과 동일함을 확인함으로써 정상적인 수행이 이루어짐을 확인하였다.

그림 9는 허프만 하드웨어 IP와 소프트웨어와의 연동실험 결과 문서 영상으로써 454K 바이트(Byte) 크기의 영상을 그림 9(a)의 경우 19K 바이트 그림 9(b)의 경우 36K 바이트로 각각 JBIG2 압축 후에 복원한 것이다.

5. 결 론

본 논문에서는 JBIG2의 부호화기중에서 가장 중요한 것 중 하나인 허프만 부호화기의 하드웨어 IP 설계에 관한 것이다. 허프만 부호화기는 JBIG2 부호화기의 임베디드 시스템 개발을 위해 개발이 선행되어야 하는 모듈이다. 본 논문에서는 JBIG2에 사용되는 15개 허프만 부호화 테이블을 하드웨어 메모리에 저장하고, 선택되어진 하나의 테이블에서 부호화 하고자 하는 값이 해당하는 구간의 최저값 만을 저장된 메모리로부터 읽어오도록 하여 메모리 접근을 최소화하여 고속으로 동작하는 하드웨어 IP를 설계하였다. 설계 및 검증은 Xilinx사의 ISE/EDK 를 이용하여 ML410보드에서 Virtex-4 FX60 FPGA를 이용하여 실험을 수행하였다. 하드웨어 사용량은 639 slices 정도로 실험에 사용한 전체 FPGA 용량의 2%정도를 사용하였다. 설계된 허프만 하드웨어 IP가 소프트웨어와 연동되어 한 번의 데이터가 부호화되는 평균 처리시간은 6.25μs이다. Microblaze 코어를 이용한 임베디드 시스템에서 소프트웨어로만 처리한 경우의 평균 처리시간은 65μs로 설계된 허프만 부호화기 하드웨어 IP가 10배 이상 빠른 처리속도를 보였다. 소프트웨어와의 연동 실험에서도 정상적인 동작을 수행하여 설계된 IP가 잘 동작됨을 검증하였다. 본 논문을 통해 개발된 허프만 부호화기는 JBIG2 부호화 칩의 주요 모듈로 사용이 가능하다. 앞으로 JBIG2에 필요한 심벌추출기, MMR, MQ, 비교기 모듈의 하드웨어 개발이 필요하다.

참 고 문 헌

[1] JBIG2 Committee, "JBIG2 Final Committee Draft," *ISO/IEC JTC1/SC29/WG1*, July, 1999.

[2] JBIG2 Committee, "JBIG2 File Format," *ISO/IEC JTC1/SC29/WG1*, Oct, 1997.

[3] P. G. Howard, "AT&T JBIG2 Coder Proposal," *ISO/ISE JTC1/SC29/WG1*, Feb, 1996.

[4] CCITT Rec. T.4, Standardization of Group 3 Facsimile Apparatus for Document Transmission, 1979.

[5] CCITT Rec. T.6, Facsimile Coding Schemes and Coding Control Function for Group 4 Facsimile Apparatus, 1988

[6] ITU-T Rec. T.82, Information Technology -Coded representation of Picture and Audio Information -Progressive Bi-Level Image Compression, Mar. 1993.

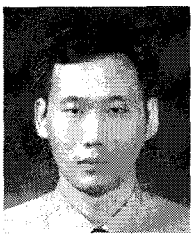
[7] http://www.adobe.com

[8] http://www.leadtools.com

[9] http://www.verypdf.com

[10] D. Pellerin and S. Thibault, *Practical FPGA Programming in C*, Prentice Hall, 2005.

[11] 김혁, Real Xilinx Process World, 엔트미디어, 2005.



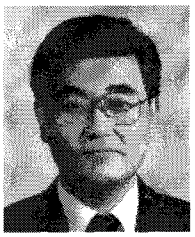
박 경 준

1999년 2월 서울산업대학교 전자공학과 학사

2001년 2월 광운대학교 전자통신공학과 석사

2001년 8월~현재 광운대학교 전자통신공학과 박사과정

관심분야 : 영상처리, 임베디드 시스템



고 형 화

1979년 2월 서울대학교 전자공학과 학사

1982년 2월 서울대학교 전자공학과 석사

1989년 2월 서울대학교 전자공학과 박사

1985년 3월~현재 광운대학교 전자통신공학과 교수

관심분야 : 영상통신, 임베디드 시스템, JBIG2