技術論文 DOI:http://dx.doi.org/10.5139/JKSAS.2011.39.9.896

DSP와 FPGA의 Co-design을 이용한 원격측정용 임베디드 JPEG2000 시스템구현

유제택*, 현명한*, 남주훈**

A Co-design Method for JPEG2000 Video Compression System in Telemetry using DSP and FPGA

Jae Taeg Yu*, Myung Han Hyun* and Ju-Hun Nam**

ABSTRACT

In this paper, a co-design method for JPEG2000 video compression system using DSP and FPGA is presented. By profiling the complexity of JPEG2000 algorithm, it is noticed that a MQ-coder is the most complex part. Thus, we implement the MQ-coder on FPGA for the parallel processing using VHDL to reduce the complexity. In order to verify the performance of the MQ-coder, JBIG2 standard test vector and images are used. The experimental results show that the proposed MQ-coder enhances the processing time approximately 3 times compared with the previous software MQ-coder.

초 록

본 논문에서는 차세대 영상 압축 표준으로 주목받고 있는 JPEG2000 알고리즘을 유도탄 원격측정용 영상압축모듈 임베디드 시스템(embedded system)에서 효율적으로 구현하기 위한 DSP와 FPGA co-design 방법을 제안한다. DSP와 함께 FPGA에서 co-processing 할 부분은 JPEG2000 알고리즘 가운데서 계산량이 많으면서도 FPGA 상에 구현하기 적합한 알고리즘인 MQ-코더 부분을 소프트웨어 profiling 작업을 거쳐 선정하였고 VHDL 언어를 사용해서 병렬 처리에 적합하도록 설계하였다. 구현한 MQ-코더의 성능을 검증하기 위하여 JBIG2 표준 테스트 벡터 및 실제 영상을 사용하였다. 실험결과 본 논문에서 제안한 MQ-코더는 기존 소프트웨어 코더보다 약 3배 정도의 압축속도를 향상 시켰다.

Key Words : JPEG2000(영상압축표준), DSP(디지털신호 처리기), FPGA(현장 프로그래머블 게이트 어레이), Co-design(융합 설계)

Ⅰ. 서 론

멀티미디어 사용이 증가함에 따라 인터넷과 디지털 기기에 사용되는 이미지 또한 고품질의 화질이 요구된다. 정지영상 압축의 표준인 JPEG은 영상을 8×8블록으로 분할해서 DCT(Discrete Cosine Transform) 수행 후에 압축하기 때문에 압축률을 크게 할 경우 블록화 현상이 심하게 일어나는 단점을 가지고 있다.

이에 따라 새로운 표준안으로 JPEG2000이 만들어 졌는데, JPEG2000은 현존하는 표준안보다 왜곡율을 낮추고 특정 부분의 이미지 품질을 한 차원 높일 뿐만 아니라 압축률 면에서도 훨씬 더

† 2011년 4월 22일 접수 ~ 2011년 7월 27일 심사완료
* 정회원, 국방과학연구소
교신저자, E-mail: uncle0421@hanmail.net
대전광역시 유성구 유성우체국 사서함 35호
** 정회원, 단암시스템즈(주)

우수하다. 무손실과 손실압축, 큰 이미지 압축, 화소의 정밀도와 해상도의 차이에 의한 점진적 전송, 관심영역(Region of Interest) 지정 및 처리 등은 JPEG2000의 특징 중에 하나이며 응용분야는 의료영상, 인터넷 영상, 원격 탐사 영상 등과 같이 서로 다른 특성을 갖는 다양한 형태의 정지 영상에 점점 더 활용도가 높아지고 있다. 그러나 JPEG2000이 여러 장점들을 갖는 반면에 이전 JPEG 보다 훨씬 많은 연산량과 메모리를 필요로 하게 되었다[1][2][3].

본 논문에서는 위에 언급한 특징을 가진 JPEG2000 알고리즘을 여러 제약사항들을 가질 수 있는 임베디드(Embedded) 환경에서 압축을 더욱 효율적으로 할 수 있는 DSP와 FPGA co-design 방법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 JPEG2000 알고리즘을 분석하여 DSP와 FPGA co-design 방법을 제안하고, 3장에서는 제안한 DSP와 FPGA co-design 방법의 실험 결과를 보이고, 4장에서 본 논문의 결론을 맺는다.

II. DSP와 FPGA Co-design

본 장에서는 JPEG2000 알고리즘을 분석하여 어떤 부분을 FPGA로 구현하는 것이 효율적인지 분석한다[4]. 또한 구현된 하드웨어 구조를 설명하고 MQ-코더를 VHDL(Very High Speed Integrated Circuit Hardware Description Language) 언어를 사용해서 FPGA로 구현하는 것에 대해서 자세히 설명한다.

2.1 하드웨어 시스템

본 논문에서 사용된 전체 시스템의 구성은 그림 Fig. 1과 같다. 영상 입력의 경우 PCI나 JTAG을 통해 PC에서 DSP로 영상 데이터를 전송하거나 외부 카메라로 NTSC 영상을 입력할 수 있다. PC에서 직접 입력하게 되는 경우 디지털 형태의 데이터이기 때문에 ADC(Analog to Digital Convert) 작업 없이 바로 영상 처리를 할 수 있다. 그러나 외부 카메라로부터 NTSC 영상을 입력하게 되면 비디오 디코더를 통해 디지털 형태인 YUV422 포맷으로 디지털화 하고, FPGA, FIFO를 거쳐서 DSP에 입력한다. 입력된 영상데이터는 DSP를 통해 영상 압축된 후, 이 압축된 영상들은 PCI를 통해 PC로 전송되거나, FIFO, FPGA, 비디오 인코더를 거쳐 NTSC로 출력된다.

영상 입력을 받은 후 영상 데이터는 DSP와 FPGA로 나누어져 처리 되고 DSP와 FPGA 사이의 통신은 DSP의 외부 메모리 인터페이스를 사용하여 버퍼를 통해 이루어진다. 전체 시스템은

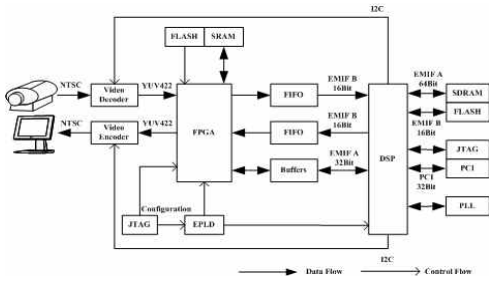

Fig. 1. 시스템 전체 구성

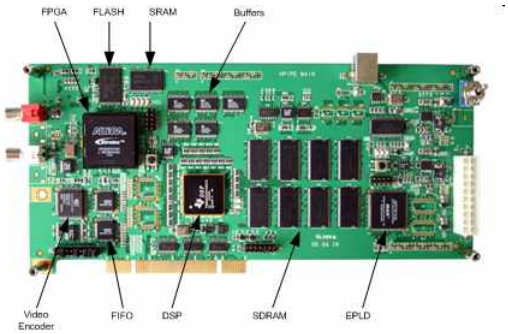

Fig. 2. 구현 된 보드사진

EPLD에 의해 컨트롤 되고 비디오 디코더와 인코더, 전체 시스템 PLL의 초기화는 DSP의 GPIO를 이용해 I2C 통신으로 설정된다. 시스템의 전원은 ATX 전원을 사용하고, PC와 PCI로 통신하기 위해 PCI Slot Type으로 PCB를 만들었다. Fig. 2는 실제 구현된 보드의 사진이다.

2.1.1 DSP(Digital Signal Processor)의 구성

본 논문에서는 영상 처리 분야에서 뛰어난 성능을 발휘하는 DSP를 주 프로세서로 사용하였다. TMS320C6416은 Texas Instruments 社의 고정소수점(Fixed point) DSP이며, 핵심 연산 부분은 VelociTI.2 Advanced Very Long Instruction Word(VLIW) Core를 사용해서 최대 8개의 명령을 한 사이클에 수행할 수 있다[5]. 현재 최대 1GHz에서 8000MIPS, 1ns의 cycle time으로 동작하며, 본 논문에서는 600MHz, 4800MIPS, 1.67ns cycle time을 가진 DSP를 사용하였다. TMS320C64x는 1MB의 내부 메모리를 가지고 있으며 최대 1280MB의 외부 메모리를 이용할 수 있다. DSP는 두 채널의 외부 메모리 인터페이스(EMIF, External Memory InterFace)를 가지고 있으며, EMIF-A는 64bit wide, EMIF-B는 16bit wide이고 최대 133MHz로 동작한다. 제안방법에

서는 FIFO와의 동기를 위해서 50MHz로 동작시켰다. DSP와 주변 장치들과의 메모리의 구성은

Table 1. TMS320C6416 Memory Map

NAME	SIZE	CE	Allocated Space(Hex)	Used Space (Hex)
Internal RAM	1MB	-	0000 0000 - 000F FFFF	0000 0000 - 000F FFFF
Internal Registers	-	-	0180 0000 - 0200 0033	0180 0000 - 0200 0033
McBSP Data	256MB	-	3000 0000 - 3FFF FFFF	3000 0000 - 3FFF FFFF
SDRAM EMIF-B	4M×16 (8MB)	CEB 0	6000 0000 - 63FF FFFF	6000 0000 - 601F FFFF
FLASH	1M×8×2Pages (2MB)	CEB 1	6400 0000 - 67FF FFFF	6000 0000 - 601F FFFF
Video FIFO	1×16 (2Bytes)	CEB 2	6800 0000 - 6BFF FFFF	6800 0000 - 6800 0001
FASTPort B FIFO	1×16 (2Bytes)	CEB 3	6C00 0000 - 6FFF FFFF	6C00 0000 - 6C00 0001
SDRAM EMIF-A	32M×64 (256MB)	CEA 0	8000 0000 - 8FFF FFFF	8000 0000 - 8FFF FFFF
Support Logic	Variable	CEA 1	9000 0000 - 9FFF FFFF	9000 0000 - 9FFF FFFF
TI Cross Platform CEA	512K×32×2Pages (4MB)	CEA 2	A000 0000 - AFFF FFFF	A000 0000 - A03F FFFF
TI Cross Platform CEB	512K×32×2Pages (4MB)	CEA 2		A040 0000 - A07F FFFF
FASTPortA FIFO	1×32 (4Bytes)	CEA 3		B000 0000 - B000 0007

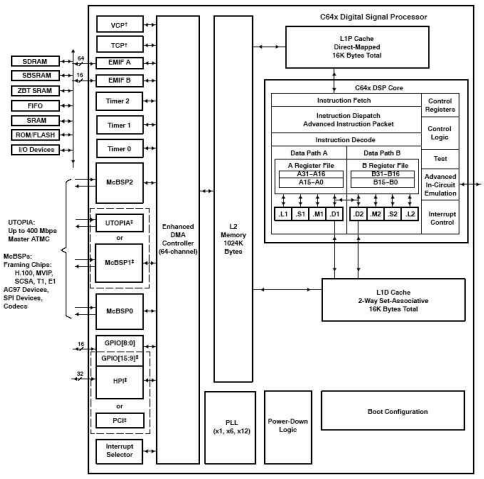

Fig. 3. TI社TMS320C6416의 블록다이어그램

Table 1과 같고 Fig. 3은 TMS320C6416 Core 및 주변 인터페이스들을 기능별 블록 다이어그램으로 표현한 그림이다.

2.1.2 FPGA(Field Programmable Gate Array)의 구성

제안방법에서는 DSP에서 영상 코덱을 처리하면서 병목 현상이 일어나는 부분의 처리 속도를 향상시키기 위해 FPGA를 사용하였다. ALTERA社의 EP1S20F672C6은 Stratix Device FPGA로써

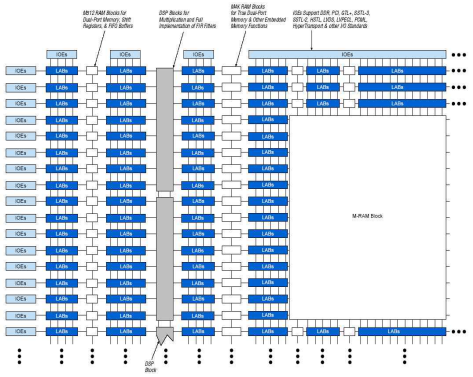

Fig. 4. ALTERA社 Stratix의 기본 구조

18,460개의 LE(Logic Element)를 가지고 있으며 총 426개의 I/O를 사용할 수 있다. 또한 총 1,669,248Bit의 내부 메모리를 가지고 있으며 영상 처리나 기타 신호 처리를 빠르게 할 수 있는 DSP 블록을 내장하고 있다. Fig. 4는 본 논문에서 사용한 FPGA인 Stratix의 기본 구조를 나타내고 있다[6].

2.2 소프트웨어 분석

소프트웨어 분석은 JPEG2000 알고리즘을 분석하여 어떤 부분에서 가장 시간이 많이 소모되는지 확인하고 어떤 부분을 FPGA로 구현하였을 때 가장 효율적인지 결정하는 데에 목적이 있다.

2.2.1 Profiling

Profiling은 여러 모듈로 코드가 구성되어 있을 경우 각 부분별로 소모되는 시간 등을 측정할 수 있는 일반적인 방법이다. 본 논문에서는 Visual C 6.0의 profiling tool을 사용해서 JPEG2000 알고리즘을 분석하였다. Profiling의 목적은 앞서도 설명했지만, 소프트웨어의 어떤 부분이 시간을 얼마만큼 소모하는지 분석하여 FPGA로 구현할 부분을 결정하는데 도움을 주는 것이다. Profiling에 사용된 조건은 Table 2와 같다.

Table 2. Profiling 환경

Processor	Intel Pentium4
Processor clock speed	2.4GHz
RAM	1GB
OS	Windows XP
Profiling tools	Visual C++ 6.0
Image	Lena
Image size	512 x 512 (24bit)
Compression type	100:1 Encoding

2.2.2 Profiling 결과

Table 3은 Profiling 결과의 일부를 보여준다. Table 3의 결과에서 실행시간의 0.5% 이하를 차지하는 모듈은 생략하였다. 이 결과를 보면 JPEG2000 소프트웨어에서 주로 시간을 많이 소모하는 모듈은 Coefficient Bit Modeling, Arithmetic Entropy Encoding (MQ-코더)부분과 Wavelet Transform 부분임을 알 수 있다.

Table 3. Profiling 결과

```
Profile: Function timing, sorted by time
Program Statistics
------------------
    Total time: 3650.468 millisecond
    Time outside of functions: 0.427 millisecond
    Call depth: 19
    Total functions: 562
    Total hits: 4666746
    Function coverage: 42.3%
    Overhead Calculated 4
    Overhead Average 4
Module Statistics for jasper.exe
--------------------------------
    Time in module: 3650.041 millisecond
    Percent of time in module: 100.0%
    Functions in module: 562
    Hits in module: 4666746
    Module function coverage: 42.3%

      Func            Func+Child          Hit
      Time     %      Time     %          Count  Function
-------------------------------------------------------------------
   487.727  13.4   954.136  26.1           1100  _jpc_encsigpass (jpc_t1enc.obj)
   483.330  13.2   483.372  13.2        1477620  _jpc_mqenc_codelps (jpc_mqenc.obj)
   425.138  11.6   425.182  11.6        1398151  _jpc_mqenc_codemps2 (jpc_mqenc.obj)
   297.762   8.2   580.362  15.9           1100  _jpc_encrefpass (jpc_t1enc.obj)
   245.512   6.7   405.495  11.1           1310  _jpc_encclnpass (jpc_t1enc.obj)
   237.664   6.5   420.036  11.5             30  _jpc_ft_analyze (jpc_qmfb.obj)
   230.568   6.3   230.568   6.3         786432  _inttobits (jas_image.obj)
   188.546   5.2   423.428  11.6           1536  _jas_image_writecmpt (jas_image.obj)
   184.180   5.0   184.180   5.0         786432  _bitstoint (jas_image.obj)
   180.912   5.0   180.912   5.0           5952  _jpc_qmfb1d_split (jpc_qmfb.obj)
   171.475   4.7   360.702   9.9              3  _jas_image_readcmpt (jas_image.obj)
    76.907   2.1    76.931   2.1              1  _jpc_enc_pi_create (jpc_t2enc.obj)
    54.150   1.5   479.546  13.1              1  _pnm_getdata (pnm_dec.obj)
    46.639   1.3    47.065   1.3            192  _jpc_tagtree_create (jpc_tagtree.obj)
    32.322   0.9  3132.966  85.8              1  _jpc_enc_encodemainbody (jpc_enc.obj)
    28.702   0.8  2012.784  55.1            210  _jpc_enc_enccblk (jpc_t1enc.obj)
    26.977   0.7    26.977   0.7          11815  _jas_malloc (jas_malloc.obj)
    25.683   0.7    25.683   0.7          11815  _jas_free (jas_malloc.obj)
```

이 중 Coefficient Bit Modeling과 Arithmetic Entropy Encoding 부분이 가장 시간을 많이 소모하였는데 Arithmetic Entropy Encoding 부분이 소프트웨어의 다른 모듈들과 독립적으로 구성되어 있어서 이 부분을 VHDL로 FPGA에 구현하기로 결정하였다.

2.3 VHDL 구현

본 절에서는 FPGA에 구현하기로 선택된 MQ-코더에 대해서 자세히 설명하고 이것을 VHDL 언어를 사용해서 FPGA로 구현하는 것에 대해서 설명한다.

2.3.1 이진산술코딩, MQ-코더

JPEG2000에 사용된 산술코딩 방법은 컨텍스트 기반 적응적 이진 산술코더(Context-based adaptive binary arithmetic MQ-coder)라고 불린다. 이것은 JBIG2에 사용된 코더와 같다[7].

MQ-코더는 Table 4와 같이 각 심볼의 확률값을 비롯해서 총 4가지의 정보를 필요로 한다. I(CX)는 Qe 값의 현재 인덱스를, Qe(I(CX))는 확률값을, NMPS(I(CX)) / NLPS(I(CX))는 MPS/LPS 재정규화(renormalization)를 위한 다음 인덱스를, 그리고 SWITCH(I(CX))는 MPS(CX)의 구간 변경이 필요한 지를 나타내는 플래그 역할을 한다. 인코딩과 디코딩에는 같은 표를 사용한다.

MQ-코더의 인코딩 흐름도는 Fig. 5와 같다. Decision bit(D)와 More Probable Symbol (MPS(CX))의 값에 따라서 "CodeMPS"와 "CodeLPS"가 선택된다. 모든 심볼들이 처리된 뒤 압축된 최종 코드워드를 만들어내기 전에 가능한 많은 이진수 1 비트들을 register C에 채워 넣기 위해서 "FLUSH register" 과정이 수행된다.

Table 4. 이진산술코딩을 위한 Lookup Table

Index	Qe	NMPS	NLPS	SWITCH
0	0x5601	1	1	1
1	0x3401	2	6	0
2	0x1801	3	9	0
3	0x0AC1	4	12	0
4	0x0521	5	29	0
5	0x0221	38	33	0
6	0x5601	7	6	1
7	0x5401	8	14	0
8	0x4801	9	14	0
9	0x3801	10	14	0
10	0x3001	11	17	0
11	0x2401	12	18	0
12	0x1C01	13	20	0
13	0x1601	29	21	0
...	...	...	...	...
39	0x0085	40	37	0
40	0x0049	41	38	0
41	0x0025	42	39	0
42	0x0015	43	40	0
43	0x0009	44	41	0
44	0x0005	45	42	0
45	0x0001	45	43	0
46	0x5601	46	46	0

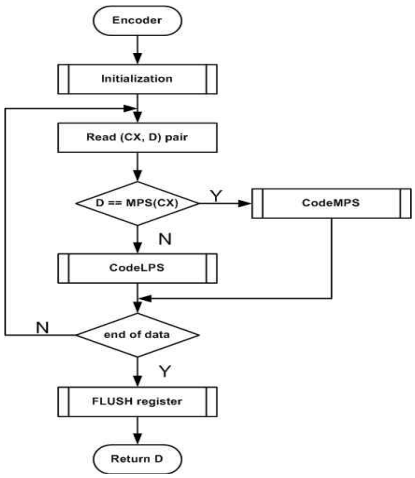

Fig 5. MQ-코더 인코딩 흐름도

2.3.2 JPEG2000 MQ-코더 분석

JPEG2000 표준에 나와 있는 인코더의 기본적인 입력과 출력은 Table 5와 같고, JPEG2000의 MQ-코더는 다음과 같은 4개의 task로 구성되어 있다. 1. Initialize(인코더의 상태를 초기와 시킨다), 2. Set Context(현재 인코더의 컨텍스트를 정한다), 3. Encode Bit(입력 데이터의 다음 비트를 인코딩한다), 4. Flush(입력 비트 스트림의 끝을 알린다). 그리고 JPEG2000은 'default'와 'predictable' 두 가지 종류의 종료를 지원한다.

Table 5. MQ-코더 data inputs/outputs

Data	설 명
Input Bit	Coefficient bit modeling의 결과로 만들어진 심볼들은 한 비트씩 인코딩된다.
Context	실제로 인코딩되는 비트외에 인코더"context"라고 부른다. "context"는 아래 두 가지로 구성된다. *Index : 표 3.3의 인덱스로 JPEG2000 FCD의 표 C-2의 확률값이다. *More-Probable Symbol (MPS) : 인코더는 항상 0 또는 1을 선택한다.
Output Byte	입력을 bit 단위로 받고 출력은 바이트 단위 나옴

2.3.3 Controller design

JPEG2000 알고리즘 코드를 분석하여 이를 VHDL로 설계하는데 필요한 register는 Table 6과 같다. 앞서 JPEG2000에서 구현된 task 들을 구현하는데 있어서 주의해야 될 사항은, VHDL 언어의 장점을 발휘해서 모든 데이터 패스가 병렬처리 되도록 설계하는 것이다. 이러한 점들을 고려해서 최종적으로 완성한 encoder entity의 입출력은 Table 8과 같다.

Table 6. Registers

Register	Width (bits)	설 명
C	32	Code register. 압축 데이터를 생성
A	32	산술코딩의 Interval register
CT	5	Counter register. A와 C 레지스터의 shift 횟수
MPS	1	현재 More Probable Symbol
Index	6	현재 look-up table 인덱스
B	8	출력 byte buffer. 다음 압축바이트 저장
TempC	32	임시 C register
K	5	Flush task에서 'predictable termination'에 사용됨

Table 7. MQ-코더 entity의 입출력

Input/Output	Width(bits)	설 명
nreset	1	Reset signal
clock	1	System clock
In_mps	1	More Probable Symbol
In_index	6	Look-up table index
In-bit	1	인코더 입력 비트
predictTerm	1	default termination -> 0 predictable termination -> 1
Init_do	1	인코더 초기화 task
Set_Context_do	1	Set context task
Encode_Bit_do	1	인코딩 task
Flush_do	1	Flush task
byte_out_reg	8	압축 데이터
enc_ready	1	인코더 idle -> 1 인코더 busy -> 0
byte_ready	1	새 데이터가 byte_out_reg에 저장될 때 1 클럭동안 "1"
c_reg_out	32	인코더의 내부 상태 표시
a_reg_out	32	
ct_reg_out	5	
mps_reg_out	1	
index_reg_out	6	

III. 실험결과

VHDL로 설계된 MQ-코더를 Altera 社의 QuartusII v5.1을 사용해서 컴파일, 합성 그리고 fitting 까지 성공적으로 수행하였다. 디바이스는 Altera 社의 StratixII를 사용하였다. 합성결과 구현된 MQ-코더의 최대 동작 주파수는 Fig. 6에서 보는바와 같이 112MHz이다.

JPEG2000 표준에는 MQ-코더의 성능을 검증하기 위한 테스트 벡터가 없기 때문에 구현된 MQ-코더의 성능을 평가하기 위해 일차적으로 JBIG2 테스트 벡터를 사용하여 검증하였다. JPEG2000에 사용된 MQ-코더와 JBIG2에 사용된

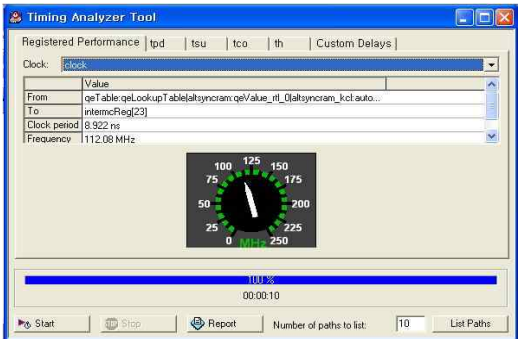

Fig. 6. 타이밍 분석 결과

MQ-코더는 같은 코더이기 때문에 JBIG2 테스트 벡터를 사용했을 때 동일한 결과가 나와야 된다. 검증을 위한 시뮬레이터는 Modeltech 사의 ModelSim 6.0을 사용했으며 시뮬레이션 결과는 Fig. 7과 같다. 시뮬레이션 결과 출력 레지스터 byteoutreg 의 값 "110111" 은 JBIG2 테스트 벡터에서 정의된 값을 가진 레지스터 expectbyte 의 값 "110111" 과 일치하는 것을 볼 수 있다 [8][9]. 총 수행 클럭은 1,118 클럭이며 110MHz 클럭으로 동작했을 때 예상 수행시간은 약 0.1usec 이다.

두 번째 검증은 제2장 profiling에 사용된 512×512 (24bits) 크기의 Lena 영상을 사용하였다. profiling 결과에서 MQ-코더에 해당되는 주요 부분은 codelps와 codemps2 부분이다. Table 8와 같이 레퍼런스 소프트웨어로 구현된 MQ-코더의 총 수행시간은 약 908.47ms 로 추정된다.

Table 8. MQ-코더 수행시간(소프트웨어구현)

codelps(ms)	codemps2(ms)	총 소요시간(ms)
483.33	425.14	908.47

Table 9. 구현된 MQ-코더의 성능

Num Clock Cycles	Maxium Freq.	Estimated Time using VHDL (ms)	% of Software Proc. Time
31,177,028	110MHz	283.43	31.20%

동일한 512×512 Lena 영상을 VHDL로 구현된 MQ-코더를 사용하여 수행 시간을 측정하였고, Fig. 8은 시뮬레이션 수행 결과를 나타낸다. 시뮬레이션 결과에 따른 MQ-코더의 성능 예측은 Table 9와 같다. 하드웨어로 구현된 MQ-코더의 성능은 110MHz 클럭을 사용했을 때, 약 283.43ms 의 시간이 소요될 것으로 예측되며 소프트웨어로 구현된 MQ-코더 보다 약 3배 정도 처리 속도가 향상되는 것을 알 수 있다.

Fig. 9는 압축하기 전의 512×512 크기의 Lena 영상이며, Fig. 10은 512×512 크기의 Lena 영상을 100대 1로 압축했을 때의 결과 영상이다.

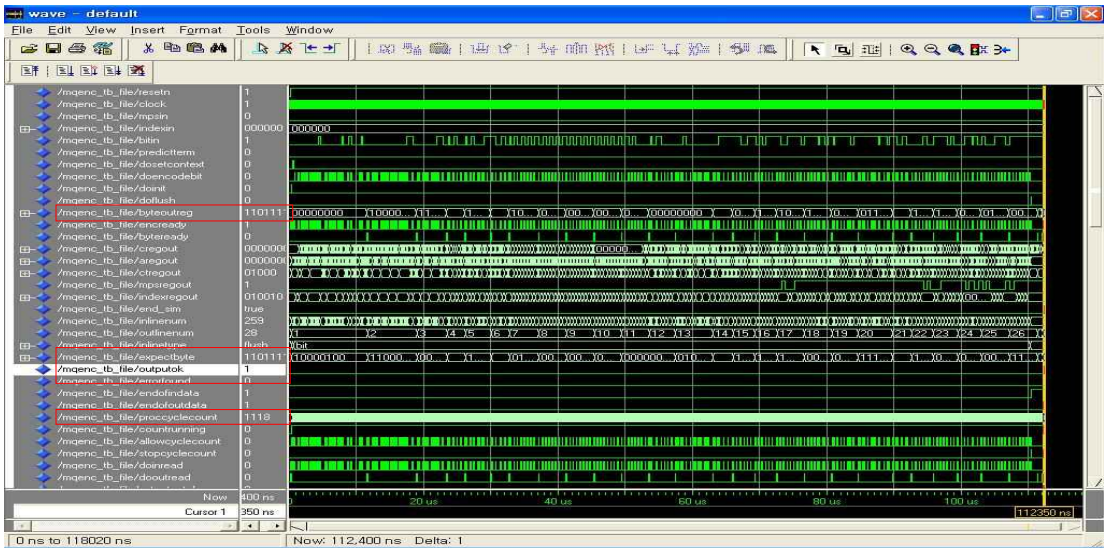

Fig. 7. JBIG2 테스트 벡터 시뮬레이션 결과

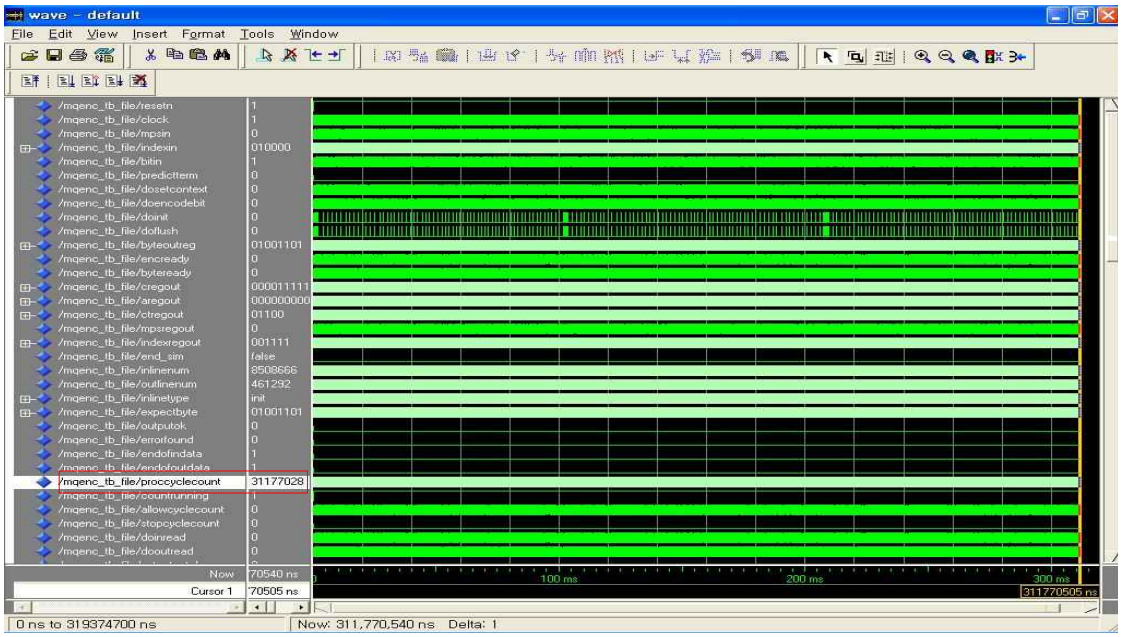

Fig. 8. Lena 영상 시뮬레이션 결과

Fig. 9. 원 영상

Fig. 10. 압축결과 영상

IV. 결 론

본 논문에서는 차세대 영상 압축 표준으로 주목받고 있는 JPEG2000 알고리즘을 임베디드 시스템(embedded system)에서 효율적으로 구현하기 위한 DSP와 FPGA co-design 방법을 제안하였다. 제안방법을 구현하기 위해 TMS320C6416 DSP와 StratixII FPGA를 사용하였고, DSP와 함께 FPGA에서 co-processing할 부분은 JPEG2000 알고리즘 가운데서 계산량이 많으면서도 FPGA상에 구현하기 적합한 알고리즘을 소프트웨어 profiling 작업을 거쳐 선정하였다. 그 결과 JPEG2000의 여러 모듈 중에서 MQ-코더 부분을 선정하였고, VHDL 언어를 사용해서 병렬 처리에 적합하도록 설계하였다. 구현한 MQ-코더의 성능을 검증하기 위해서 JBIG2 표준 테스트 벡터와 실제 영상을 사용하였다. 실험결과 본 논문에서 제안한 MQ-코더는 기존 소프트웨어 코더보다 약 3배 정도의 압축속도 향상을 보여주었다.

본 논문에서 사용한 DSP 와 FPGA co-processing 시스템의 향후 성능 향상을 위해서는 DSP와 FPGA간의 고속 bridge 개발, FPGA 내부의 DSP 블록과 내부 메모리의 효율적 활용, FPGA 내부 알고리즘의 병렬처리 최적화에 대한 연구가 필요하다.

참고문헌

1) M.D. Adams and F. Kossentini, "JasPer: a software-based JPEG-2000 codec implementation", in: Proceedings of the IEEE International Conference on Image Processing, Vancouver, CA, Sept. 2000.

2) M. D. Adams, H. Man, F. Kossentini, and T. Ebrahimi, "JPEG2000: the next generation still image compression standard", ISO/IEC JTC1/SC29/WG1 N1734, June 2000.

3) M. Rabbani and R. Joshi, "An overview of the JPEG2000 still image compression standard", Signal Processing: Image Communication, vol. 17, no. 1, pp. 3-48, Jan. 2002.

4) JasPer Project Home Page, http://www.ece.ubc.ca/~mdadams/jasper/.

5) "TMS320C6416 Fixed-Point Digital Signal Processors", Texas Instruments, http://focus.ti.com/docs/prod/folders/print/tms320c6416.html.

6) "Stratix II Device Handbook", Altera, http://www.altera.co/products/devices/stratix2/overview/st2-overview.html.

7) T. Acharya, P. S. Tsai, "JPEG2000 Standard for Image Compression", John Wiley & Sons, Inc., 2005.

8) ISO 14492 FCD, "Information Technology - Coded Representation of Picture and Audio Information - Lossy/Lossless Coding of Bi-Level Image (JBIG2)", July 1999.

9) "ModelSim SE Manual", Mentor Graphics, http://www.model.com.