DBpia

논문 15-40-12-19 The Journal of Korean Institute of Communications and Information Sciences '15-12 Vol.40 No.12
http://dx.doi.org/10.7840/kics.2015.40.12.2469

FPGA를 이용한 하드웨어 기반 고성능 XML 파싱 기법

이 규 희◆°, 서 병 석*

Hardware-Based High Performance XML Parsing Technique Using an FPGA

Kyu-hee Lee◆°, Byeong-seok Seo*

요 약

다양한 웹 서비스들은 서비스의 제공을 위해서 구조화된 표준문서인 XML(eXtensible Markup Language)을 널리 사용하고 있으며, 모바일 환경에서의 전자문서 및 전자서명 그리고 메일시스템에서도 XML이 사용되고 있다. XML을 사용하기 위해서는 문서의 파싱이 요구되며, 이는 XML 처리에서 가장 계산 집중적 작업이다. 따라서, XML 파싱 성능을 높이기 위해 하드웨어 기반의 파서들이 제안되어 성능 향상에 초점을 맞추고 있지만 실제 파싱 기법들에 대한 연구는 거의 이루어지지 않았다. 본 논문에서는 파서의 종류와 상관없이 사용될 수 있는 고성능 XML 파싱 기법을 제안하고 FPGA를 이용하여 파서를 설계하여 검증하였다. 제안된 파싱 기법은 상태머신 대신에 엘리먼트 분석기들을 사용하며 다중바이트 단위 엘리먼트 매칭을 수행한다. 제안된 파싱 기법은 CPB 항목에서 약 2~4배의 소비 클록을 감소시켰으며 파싱 이전에 전처리작업을 요구하지 않는다. 다른 파서들과 비교하여 제안된 파서는 약 1.33~1.82배 속도를 향상시켰다. 따라서, 제안된 파싱 기법은 실시간 XML 파싱이 가능하며 일반적인 XML 파서들에서도 적용할 수 있는 적합한 구조를 갖는다.

Key Words : XML hardware parser, multibyte parsing, parsing technique, element analyzer, FPGA

ABSTRACT

A structured XML has been widely used to present services on various Web-services. The XML is also used for digital documents and digital signatures and for the representation of multimedia files in email systems. The XML document should be firstly parsed to access elements in the XML. The parsing is the most compute-instensive task in the use of XML documents. Most of the previous work has focused on hardware based XML parsers in order to improve parsing performance, while a little work has studied parsing techniques. We present the high performance parsing technique which can be used all of XML parsers and design hardware based XML parser using an FPGA. The proposed parsing technique uses element analyzers instead of the state machine and performs multibyte-based element matching. As a result, our parsing technique can reduce the number of clock cycles per byte(CPB) and does not need to require any preprocessing, such as loading XML data into memory. Compared to other parsers, our parser acheives 1.33~1.82 times improvement in the system performance. Therefore, the proposed parsing technique can process XML documents in real time and is suitable for applying to all of XML parsers.

◆° First and Corresponding Author : Sangji Youngseo College, Department of National Defense Communication Engineering, khlee@sy.ac.kr, 정회원

* Sangji Youngseo College, Department of National Defense Communication Engineering, seobs@sy.ac.kr, 정회원

논문번호 : KICS2015-10-329, Received October 8, 2015; Revised November 26, 2015; Accepted November 26, 2015

www.dbpia.co.kr

Ⅰ. 서 론

다양한 웹 서비스들은 구조화된 표준 문서인 XML(eXtensible Markup Language)을 이용하여 사용자에게 서비스를 제공한다. XML 기반의 메시지를 전달하는 웹 서비스 방식에는 SOAP(Simple Object Access Protocol)와 REST(Representional State Transfer)가 있으며[1], 모바일 환경에서의 전자문서 및 전자서명이나 전자메일 시스템에서의 멀티미디어 파일의 표현에도 널리 사용되고 있다[2-4].

XML을 이용하기 위해서 문서의 파싱작업이 요구되는데, 이는 XML의 사용에서 가장 계산 집중적 처리량을 요구한다. 파싱은 이벤트 기반 SAX (Simple API for XML)[2] 파서(스트리밍 파서)와 DOM (Document Object Model) 기반 파서가 널리 사용된다. DOM 파서는 문서 전체를 메모리에 적재한 후 파싱을 수행하기 때문에 접근이 용이하지만 전처리 작업이 요구되며 메모리 사용량이 증가한다. 반면에 스트리밍 파서인 이벤트 기반 파서들은 XML 문서가 입력되는 동시에 파싱을 수행하기 때문에 전처리 작업 및 추가적인 메모리 할당이 필요하지 않다. 따라서, 스트리밍 XML 파서들은 제한적 자원을 갖고 고속의 파싱이 요구되는 시스템들에 적합한 구조이다.

고속의 파싱 성능을 내기 위해 하드웨어 기반 XML 파서들이 제안되고 있으며, CAM (Content Addressable Memory) 또는 재구성이 가능한 FPGA (Field Programmable Gate Arrays) 등이 주로 사용되고 있다. 일반적으로 CAM은 한 클록에 비교 결과를 출력하는 고성능을 갖지만, 가격이 비싸고 높은 전력을 소비하는 단점을 갖는다. 반면에 FPGA는 gate들의 집합으로 구성되어 있어 파서의 설계 및 구조 변경이 용이하지만, CAM과 비교하여 상대적으로 낮은 성능을 갖는다.

본 논문에서는 FPGA를 이용하여 하드웨어 파서를 설계하고 CAM에 상응하는 성능을 갖도록 다중바이트 단위 파싱을 수행하고, 상태머신 대신에 엘리먼트 단위로 분석할 수 있는 인코딩 기법 및 분석기를 제안한다. 또한, 실시간으로 입력되는 XML 문서를 처리할 수 있도록 하여 DOM 기반 파서 뿐 만 아니라 스트리밍 파서에서도 적용될 수 있는 일반적인 파싱 기법을 제안한다. 제안된 파서에서는 XML의 구조를 검증하는 Well-formed 검사를 수행하며, 엘리먼트 유효성 및 속성들을 정의하는 DTD (Document Type Definition)나 제한점들을 표현한 XMLSchema 등은 고려하지 않고 전형적인 XML 문서의 파싱 기술에 대하여 집중한다.

본 논문의 2장에서는 XML 파서들에 대한 관련 연구들을 살펴보고, 3장에서는 제안된 구조의 하드웨어 구조와 동작을 기술한다. 4장에서는 제안된 스트리밍 XML 하드웨어 파서의 성능을 평가하고 5장에서 결론을 맺는다.

Ⅱ. 관련연구

실시간으로 입력되는 XML 문서의 파싱 및 검색을 위해서 스트리밍 파서들이 널리 사용되고 있다. 스트리밍 파서는 입력과 동시에 파싱을 수행하기 때문에 고속으로 동작하지만, 이미 파싱된 XML 엘리먼트의 재접근이 어려운 단점을 갖는다. 스트리밍 파서의 단점을 보완하기 위해 DOM 트리 구조를 변형한 StrexTree[5] 파서가 제안되었다. StreXTree 파서는 DOM 트리에서 단말 노드가 되는 엘리먼트 세트(시작엘리먼트, 컨텐트, 종료엘리먼트)를 상위 엘리먼트의 자식 노드 대신에 형제 노드로 연결하여 검색 속도의 성능 향상을 이루었다. 그렇지만, 여전히 단일 바이트 단위로 XML 문서를 파싱하기 때문에 성능에 제한점을 갖는다.

스트리밍 파서의 단점인 엘리먼트의 재접근과 유연성을 위해 DOM 기반 파서들이 널리 사용되고 있다. DOM 기반 파서들은 XML의 파싱전에 문서를 메모리에 적재하는 전처리 작업들이 요구된다. 따라서, 파싱의 전체 성능이 저하될 수 있다. 일반적으로 XML 파서의 성능에 대한 정량적 평가는 이전 연구에서 제시한 CPB(cycles per byte)[6]를 사용한다.

DOM을 이용한 파서에서 XPA(XML Parsing Accelerator)[7]는 XML 파싱을 가속화하기 위하여 well-formed 검사와 DOM 트리 생성을 동시에 수행하는 파이프라인 하드웨어 구조를 제시하였다. 제시된 하드웨어 구조는 일반적인 XML들에 대하여 CPB가 1인 성능을 갖는다.

PSDXP(Parallel Speculative Dom-based XML Parser)[8] 구조는 단일바이트 단위 파싱의 성능 제한점을 해결하기 위하여 한번에 여러 바이트를 파싱할 수 있도록 DOM 기반 병렬 파서를 제안하였다. 제안된 하드웨어 파서는 병렬로 XML을 파싱하기 위하여 주어진 문서를 대략적으로 균등하게 분배하여 각각의 종속 XML을 병렬로 파싱하기 때문에 단일바이트 파서들보다 높은 CPB 성능을 달성한다. 그러나, XML 문서를 균등하게 나누어 메모리에 적재하는 전처리 과정이 요구되며, 엘리먼트 단위로 문서를 나누기 때

www.dbpia.co.kr

```
<stock>
    <stockitem stockid="1">
        <share val="763">
            <name>somecompany</name>
        </share>
        <qty>60</qty>
        <price type="USD">115</price>
    </stockitem>
</stock>
```

그림 1. XML 예제
Fig. 1. An XML example.

문에 각 종속 문서의 길이에 따른 출력 지연이 발생할 수 있다.

SCBXP(Skeleton CAM-based XML Parser)는[9] 순차 처리되는 파싱구조를 개선하기 위해 CAM을 이용하여 4바이트 단위 파싱 구조를 제안하였다. SCBXP는 4개의 FIFO를 이용하여 입력되는 XML 문서를 엘리먼트 단위로 파싱하여 미리 정의된 CAM의 TokenID들과 비교한다. CAM의 비교 결과는 상태머신으로 전송되고 엘리먼트 단위 정렬이 되지 않았거나 비교가 완료되지 않은 유효한 엘리먼트를 나중에 비교한다. 여러 개의 상태머신은 단일바이트 파싱을 수행하는데 4바이트 상태머신을 사용하는 경우 상태의 개수가 폭발적으로 증가하기 때문이다. SCBXP 구조는 4바이트 단위 파싱으로써 순차 처리하는 파서들과 비교하여 고성능을 달성할 수 있지만, XML 문서의 입력이 FIFO에서 처리 될 때까지 출력 지연이 발생한다. 또한 XML 문서에서 5개의 중첩 엘리먼트에 최적화된 하드웨어를 구성하였기 때문에, 5개 이하의 중첩 엘리먼트를 갖는 문서의 파싱에서 상태 전이에 대한 추가적인 클록을 소비한다. SCBXP는 그림 1의 예를 이용하여 XML 문서의 파싱 기법과 구조적 문법 검사를 수행하는 Well-formed 검사에 대하여 기술하였다.

본 논문에서는 순차 처리의 한계성을 해결하고 전처리 및 병합과정이 요구되지 않는 다중바이트 기반의 파싱 기법을 제안한다. 제안된 파싱 기법은 상태머신을 사용하지 않는 새로운 인코딩 기법을 적용하여 엘리먼트를 분석 할 수 있어 고성능의 XML 파싱을 수행할 수 있으며, 스트리밍 XML 파서도 지원할 수 있도록 메모리에 문서를 적재하는 전처리과정을 요구하지 않도록 한다.

Ⅲ. 다중바이트 기반 XML 하드웨어 파서

3.1 제안된 하드웨어 파서의 시스템 구조

XML 문서의 순차 파싱 및 단일 문자 단위의 파싱은 네트워크 속도의 증가로 인해 실시간 파싱의 한계성을 갖는다. 따라서, XML 파서의 고속 동작을 위해 다중바이트 기반 파싱이 요구된다. 대다수의 XML 파서들은 상태머신을 이용하여 입력되는 문자에 따른 상태 전이를 통해 파싱을 수행한다. 만일, 파싱의 속도를 향상시키기 위하여 상태머신을 이용한 다중바이트 단위 파싱을 수행한다면 입력 문자열에 대한 상태 수가 급격히 증가하여 회로 복잡도가 높아지고 병목현상이 발생할 수 있다. 제안된 파싱 기법은 FPGA를 사용하여 하드웨어 파서로 설계되며 상태머신 대신에 인코딩된 엘리먼트 정보를 이용하여 다중바이트 단위

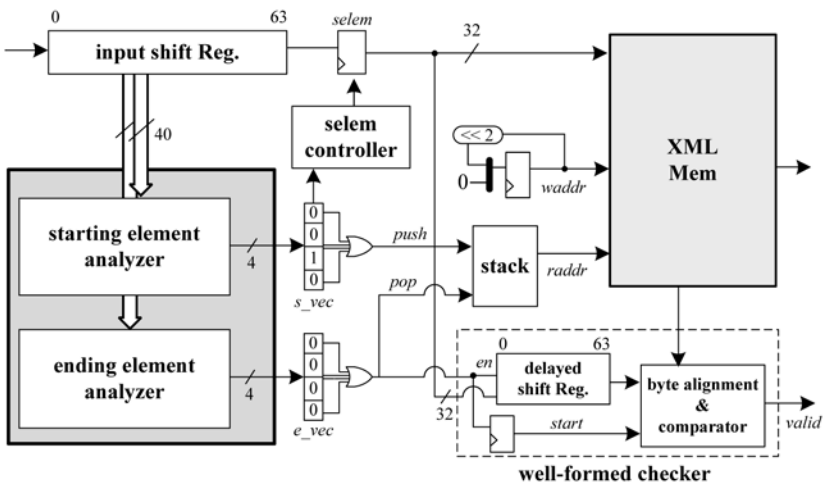

그림 2. 다중바이트 기반 XML 하드웨어 파서의 시스템 구조
Fig. 2. System architecture of multibyte-based XML hardware parser.

www.dbpia.co.kr

로 파싱한다. 그림 2는 제안하는 다중바이트 기반 하드웨어 파서의 전체 시스템 블록도이다. 일반적으로 이더넷들은 32비트 인터페이스를 가지고 있기 때문에 본 논문에서는 4바이트 단위의 새로운 파싱 기법을 적용한 하드웨어 구조를 제시한다.

그림 2의 제안된 파서는 4개 문자단위로 입력 쉬프트 레지스터에(input shift register) 입력되고, 다음 클록에서 새로운 4개 문자가 입력되면 쉬프트 동작을 수행한다. 입력 쉬프트 레지스터에서 엘리먼트 분석기로 데이터를 공급하는 동시에 시작엘리먼트 레지스터(selem)에도 저장된다. 이는 시작엘리먼트 분석기로부터 제공되는 값을 이용하여 시작엘리먼트 제어기(selem controller)에 의해 제어된다. 엘리먼트 분석기는 시작/종료 엘리먼트 분석기 두 개로 구분되며 시작엘리먼트 분석기는 입력 문자열에서 시작엘리먼트의 유무를 판별하여 출력 벡터에 값을 저장한다. 출력 벡터 값은 시작엘리먼트 제어기에서 쉬프트 회수와 스택의 push 신호로 사용되는데, 스택에는 XML 메모리에 저장되는 시작엘리먼트들의 주소를 저장한다. 마찬가지로, 종료엘리먼트 분석기는 입력 문자열에서의 종료엘리먼트를 판별하고 결과를 출력 벡터에 저장하여 스택의 pop 신호로 사용한다. 스택에서 pop된 주소는 well- formed checker의 비교를 위해 XML 메모리에서 시작엘리먼트를 인출하는데 사용된다. XML 메모리에는 문서의 시작엘리먼트 단위로 저장되기 때문에 스택을 이용하면 XML 문서의 엘리먼트 중첩구조와 계층구조를 쉽게 판별할 수 있는 장점이 있다.

3.2 엘리먼트분석기

다중바이트 단위의 XML 파싱에서 상태머신의 사용은 상태 수를 급격히 증가시켜 파서 회로의 복잡도를 높이고 성능 열화를 초래할 수 있다. 본 논문에서는 낮은 회로 복잡도를 가지며 문서의 파싱 성능을 높이기 위해서, 상태머신을 사용하지 않고 XML의 태그들과 대응되는 2비트의 인코딩 값들을 이용하여 엘리먼트들을 분석한다. 모든 엘리먼트는 2개 문자 단위로 인코딩된 2비트와 비교되는데, XML의 시작엘리먼트는 "<"로 시작하지만, 종료엘리먼트가 "</"로 구분되기 때문이다. 입력 문자에 대한 인코딩 값은 시작태그("<")는 01b, 닫힘태그("/")는 10b, 종료태그(">")는 11b, 그리고 나머지는 00b이다.

정의된 인코딩 비트들은 시작엘리먼트 분석기와 종료엘리먼트 분석기에 제공되어 그림 3과 같이 해당 입력문자열에 대하여 시작엘리먼트와 종료엘리먼트를 판별한다. 비교된 4개의 결과들은 각각 시작엘리먼트

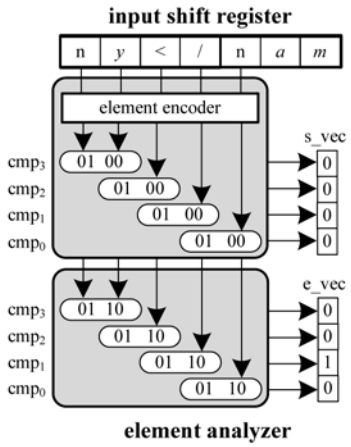

그림 3. 엘리먼트 분석기
Fig. 3. Element analyzer.

벡터(s_vec)와 종료엘리먼트 벡터(e_vec)에 저장되며, 그림 2의 스택에서 push와 pop 신호로 사용된다. 입력된 문자열들의 인코딩은 엘리먼트 인코더에서 수행되며 시작/종료엘리먼트 분석기에 동시에 제공된다.

그림 3의 각 비교기(cmp3, cmp2, cmp1, cmp0)에 저장된 2개의 2비트는 시작엘리먼트와 종료엘리먼트를 의미한다. 예를 들어 비교기 값이 "01 00"이면 시작엘리먼트를 나타내고, "01 10"이면 종료엘리먼트를 의미한다. 입력된 문자열들은 두 개의 문자 단위로 인코딩 비트들과 비교하여 엘리먼트의 종류를 판별 할 수 있다. 예를 들어, cmp0 비교기는 현재 입력 쉬프트 레지스터의 "/" 문자와 5번째 문자 "n"을 비교하여 s_vec과 e_vec에 대한 출력을 결정할 수 있다.

3.3 스택 및 XML 메모리 구조

스택에는 well-formed 검사를 위하여 XML 메모리에 저장되는 시작엘리먼트의 주소를 저장하는데, 엘리먼트분석기들로 부터 push 및 pop 신호를 입력받아 동작한다. 예를 들어, 시작엘리먼트의 출력 벡터(s_vec)에 대한 bitwise-OR 연산이 1이면 해당 시작엘리먼트가 저장되는 XML 메모리의 주소를 스택에 저장하고, 종료엘리먼트의 출력 벡터(e_vec)의 bitwise-OR 값이 1이면 스택에서 pop 신호로 사용된다. 그림 4는 그림 1의 예제 XML 문서를 사용하여 파싱할 때 스택과 XML 메모리의 동작 예를 나타내며, 엘리먼트가 저장되는 지연시간은 고려하지 않는다. XML의 파싱 시작 시간을 t라 할 때, XML 메모리에는 "<sto"가 저장되고 이 때에 s_vec의 출력이 1이 되므로 스택에 해당 메모리의 시작주소 0x00이 push된다. 시간 t+1일 때, XML 메모리에는 "ck>\0"

www.dbpia.co.kr

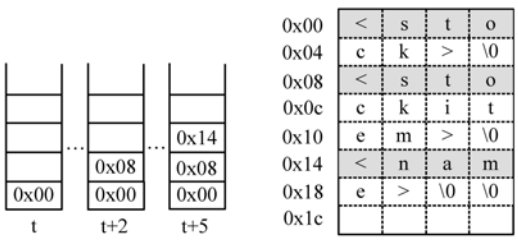

그림 4. 스택 및 메모리 사용 예
Fig. 4. example for the use of stack and memory.

이 저장되며 스택에는 변화가 없으며, 시간 t+2에서 push 동작이 다시 수행된다. 시간 t+8에서 종료엘리먼트 “</na”이 파싱되면 e_vec의 출력이 1이 되어 스택으로부터 0x14가 pop되며 well-formed 검사를 수행하기 위한 XML 메모리의 인출 주소로 사용된다.

3.4 바이트정렬기 및 4바이트 비교기

스택에서 pop된 주소는 well-formed 검사를 위한 XML 메모리의 주소로 사용된다. XML 메모리에서 인출된 주소는 시작엘리먼트 단위로 저장되어 있지만, 입력으로 들어오는 입력 쉬프트 레지스터로부터 입력되는 데이터는 임의의 위치에서 시작될 수 있기 때문에 바이트 정렬이 필요하다. 따라서, 제안된 파서 구조에서는 첫 클록에서 바이트 정렬을 수행하고 다음 클록부터는 정렬된 데이터를 이용하여 4바이트 단위로 비교된다. 시작엘리먼트는 XML 메모리에 저장될 때 마지막에 널문자(\0)를 포함하여 비교의 종료 시점을 정할 수 있도록 한다. 시작엘리먼트의 종료 시점을 표시하기 위해 사용되는 널 문자는 메모리 엔트리의 낭비를 초래할 수 있지만 이는 검색 속도를 향상시키기 위해 충분히 감내가 가능하다. 그림 5는 제안된 파서의 well-formed 검사기 구조로서 입력되는 데이터의 동기화를 위한 지연 쉬프트레지스터와 바이트정렬기 및 비교기로 구성된다.

예를 들어, 그림 4의 시간 t+5에서 종료엘리먼트 “</na”가 파싱되면 스택으로부터 주소 0x14가 pop

되고, XML 메모리에서 주소 0x14에 해당하는 시작엘리먼트 “<nam”이 인출되어 well-formed 검사를 위해 비교된다. t+5 시간에 입력된 데이터(종료엘리먼트)는 t+7에 시작엘리먼트 “<nam”과 비교되는데 이는 스택에서 한 클록 그리고 메모리 인출에서 한 클록이 소비되기 때문이다. 따라서 t+5에 입력된 데이터의 정확한 비교를 위해 지연 쉬프트레지스터를 사용하며, 비교회로의 편이성을 위해 종료엘리먼트의 “/”를 제거하여 저장한다. 이때에 입력된 데이터는 “</na”,

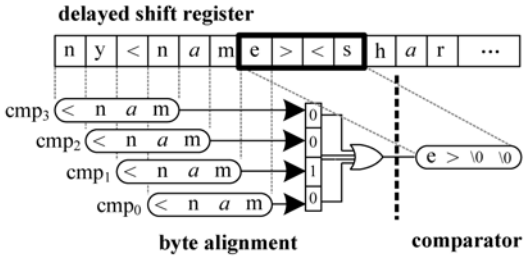

그림 5. 바이트 정렬
Fig. 5. Byte alignment.

“x</n”, “xx</”, “xxx<”과 같이 임의의 위치에 올 수 있으므로 바이트 정렬기의 4개 비교기를 통해 정렬된다. 여기에서 x는 don’t care 문자를 의미한다.

Ⅳ. 평 가

본 논문에서는 고성능 XML 파서를 완성하기 위해 하드웨어를 이용하여 다중바이트 기반 파싱 기법을 제안하였다. 제안된 파서의 XML 문서들에 대한 메모리 적재 및 파싱 완료 시간을 일반적으로 비교하기 위해, SCBXP에서 사용한 XML 문서의 전체 처리시간 산출 식을 사용하도록 한다.

$$T_{proc} = T_{parse} + T_c + (10/N) \quad (1)$$

식 (1)에서 Tparse는 XML 문서를 파싱하는 시간을 나타내고, Tc는 문서의 파싱을 시작하기 전에 소요되는 CAM 설정 시간 및 XML 문서의 메모리 적재 시간을 나타낸다. N은 XML 문서의 수를 나타내며 메모리에 적재된 XML의 엘리먼트를 읽고/쓰는 시간에 대한 지연시간 10을 포함한다. 식 (1)을 이용하여 단일바이트 단위 파서와 비교하기 위해, 식 (1)을 표 1의 SCBXP와 같은 식으로 변환하여 사용하며, 문서의 파싱시간 Tparse는 파서의 구조에 따라 약간의 차이가 있을 수 있기 때문에, 일반적인 비교를 위해 모두 같다고 가정한다. 표 1에서 StreXTree는 단일바이트 파서이며, 스트리밍 XML 문서를 지원하기 위해서 전처리작업이 요구되지 않기 때문에 Tc와 같은 요소는 포함되지 않는다. 제안된 파서는 메모리에 적재하는 전처리 작업이 요구되지 않고 파싱이 4바이트 단위로 이루어지며, 입력에서 1 클록의 지연시간을 포함하여 상대적으로 다른 하드웨어 파서들보다 효율적인 파싱 기법을 갖는다.

표 1에서 비교된 일반적인 처리시간의 비교 이외에 XML 파싱에서는 파서들의 정량적인 성능 비교를 위

www.dbpia.co.kr

표 1. XML의 파싱 처리시간 비교
Table 1. Comparison of processing time of XML.

Architecture	Processing Time	Aver. (CPB)
StreXTree[5]	$T_{proc} = T_{parse}$	1
SCBXP[9]	$T_{proc} = \lceil T_{parse}/4 \rceil + T_c + (10/N)$	0.5
Ours	$T_{proc} = \lceil T_{parse}/4 \rceil + 1$	0.25

해 CPB(Cycles per Byte)를 사용한다. 이전에 연구된 일부 XML 파서들은 평가를 위해 다른 연구들과 동일한 벤치마크 용 파일을 사용하지 않아 정량적 비교가 어려우나, 연구에서 제시된 평균 CPB를 이용하여 표 1에 함께 기술하였다. 표 1에서 보는바와 같이, 제안된 파싱기법은 평균 CPB에서 2~4배 클록을 적게 소비한다. 다른 연구들과의 보다 정량적인 비교 및 평가를 위해 식 (2)의 CPB 산출식을 사용하였다. 제안된 파서는 FPGA를 이용한 XML 전용 하드웨어로써, FPGA 합성도구에서 제시된 최대 주파수(Maximum Frequency)를 이용하였으며, 그림 6에 비교 결과를 제시하였다.

$$CPB = \frac{Frequency * Utilization(\%\ all\ CPUs / 100)}{Throughput / 8(b/B)} \quad (2)$$

그림 6에서 사용된 XML 벤치마크 파일들의 정보는 표 2에 제시하였다. 제안된 파서는 4개의 파일 모두에서 CPB가 0.25로써 XPA[7]보다 4배의 소비 클록을 감소시켰으며, PSDXPx4[8]와 같은 성능을 갖는다. 그러나, 그림 7에 제시된 결과에서 제안된 파서는 속도가 약 5.34~7.31 Gbps로써 PSDXPx4 보다 약 1.33~1.82배 빠른 파싱 성능을 갖는다. 제안된 파서는 XML 크기가 증가할 수 록 최대 주파수가 감소하여 가변적인 성능을 갖지만, 일반적으로 이더넷 인터페이스가 125MHz로 동작하는 것을 감안하면 실시간으로 입력되는 XML을 충분히 처리할 수 있는 능력을

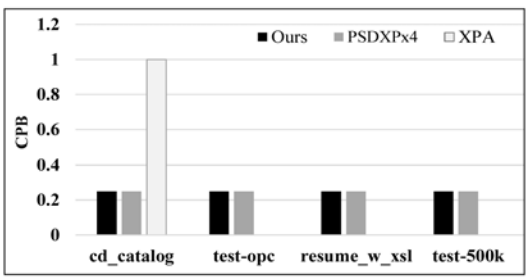

그림 6. 다른 XML 파서들과의 CPB 비교
Fig. 6. Comparison of CPB with other parsers.

표 2. 제안된 XML 파서의 합성결과
Table 2. Synthesis result of the prposed XML parser.

BenchMark	Size	ALUT	Memory (bits)	Fmax (MHz)
test-opc.xml	1.79KB	384	17,480	228.57
cd_catalog	5.66KB	387	33,872	218.87
resume_w_xsl	50.5KB	391	525,424	188.89
test-500k	500KB	480	4,195,456	166.97

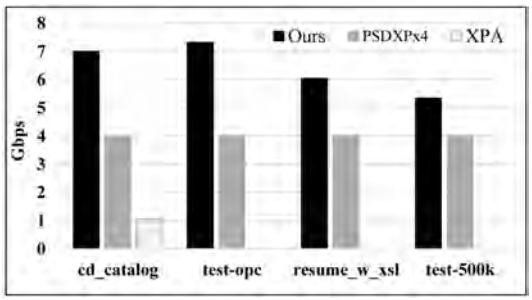

그림 7. 다른 XML 파서들과의 성능 비교
Fig. 7. Performance comparison of other parsers.

갖는다. 반면에 PSDXPx4는 최대 주파수를 125MHz로 가정하였다.

제안된 파서는 Altera의 Stratix IV FPGA 디바이스를 사용하였고 합성 도구는 QuartusII 13.0을 이용하였다. 제안된 구조의 하드웨어 합성을 위하여 그림 1의 XML 예제 대신에 일반적으로 XML 파서의 평가를 위해 사용되는 XML 벤치마크 파일들을 이용하였으며, 합성결과는 표 2에 나타내었다. 표 2에서 ALUT는 Altera의 Stratix IV FPGA를 이루는 로직 배열의 기본 단위를 의미한다.

V. 결 론

XML 파서들에서 문서의 파싱은 계산 집중적 작업으로써 파서의 성능을 결정하는 가장 중요한 요소이다. 일반적으로 스트리밍파서와 DOM 기반 파서들이 널리 사용되며 각 파서들의 특성에 따른 장단점을 이용하여 시스템에 적합한 파서를 채택하여 사용한다. 파서의 성능을 높이기 위하여 하드웨어를 이용한 고속화 연구들이 주로 수행되고 있지만, 파싱 기법에 연구들은 거의 이루어지지 않고 있다.

본 논문에서는 XML의 파싱 기법을 최적화하여 성능을 향상 시킬 수 있도록 엘리먼트 분석기를 이용한 다중바이트 단위의 하드웨어 기반 파서를 제안하였다. 제안된 구조는 다른 파서들과 비교하여 평균 CPB에

www.dbpia.co.kr

서 2~4배 소비 클록이 감소되었고 동일한 XML 벤치마크 파일을 사용한 경우, CPB에 대하여 동일하거나 4배의 소비클록이 감소하였다. 동일한 CPB 성능을 내는 경우일지라도, 제안된 파서의 전체 속도는 약 5.34~7.31 Gbps로써 약 1.33~1.82배 속도가 향상됨을 입증하였다. 또한, 제안된 파서는 파싱 이전에 전처리 작업을 요구하지 않는 방법으로 설계되어 소비클록이 감소하고 성능이 향상된 파싱 기법으로써, 모든 XML 파서들에 적용할 수 있는 일반적인 구조를 갖는다.

References

[1] G. P. Dai and Y. Wang. "XML-based structural representing method for information of things in internet of things," *Advances in Comput. Sci. Inf. Eng.*, pp. 9-15, 2012.

[2] S. M. Kang, K. T. Hwang, and N. Y. Kim, "Design and implementation of XML authoring tool for digital document on M-commerce: X-Auth," *J. KICS*, vol. 29, no. 2B, pp. 289-298, Feb. 2004.

[3] N. Y. Kim and K. T. Hwang, "The performance evaluation of XML-based digital signature system on mobile environment," *J. KICS*, vol. 29, no. 4C, pp. 570-580, Apr. 2004.

[4] S. H. Kim and K. H. Choi, "Design of an XML based multimedia mailing system," *J. KICS*, vol. 26, no. 1A, pp. 116-127, Feb. 2001.

[5] K. H. Lee and S. S. Han, "A streaming XML hardware parser using a Tree with Failure Transition," *J. KIICE,* vol. 17, no. 10, pp. 2323-2329, Oct. 2013.

[6] M. Leventhal and E. Lemoine, "The XML chip at 6 years," *Int. Symp. Process. XML Efficiently,* vol. 27, 2009.

[7] D. Zefu, N. Ni, and J. Zhu. "A 1 cycle-per-byte XML parsing accelerator," *The 18th ACM/SIGDA Int. Symp. FPGA*, pp. 199-208, Feb. 2010.

[8] M. Jianliang, et al., "Parallel speculative dom-based XML parser," *IEEE 9th Int. Conf. Embedded Softw. Syst.*, pp. 33-40, 2012.

[9] E. H. Fadi and D. Ionescu, "SCBXP: an efficient CAM-based XML parsing technique in hardware environments," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 22, no. 11, pp. 1879-1887, 2011.

이 규 희 (Kyu-hee Lee)

2007년 2월 : 연세대학교 컴퓨터정보통신공학부 학사
2009년 2월 : 연세대학교 전산학과 석사
2015년 2월 : 연세대학교 전산학과 박사
2015년 3월~현재 : 상지영서대학교 국방정보통신과 조교수
<관심분야> 네트워크보안, 임베디드시스템, FPGA

서 병 석 (Byeong-seok Seo)

2001년 2월 : 연세대학교 전산학과 학사
2008년 2월 : 연세대학교 전산학과 석사
2011년 2월 : 연세대학교 전산학과 박사 수료
2012년 3월~현재 : 상지영서대학교 국방정보통신과 조교수
<관심분야> FPGA, GPU, 클라우드컴퓨팅

www.dbpia.co.kr