특집

FPGA를 이용한 ASIC 에뮬레이터

柳 光 基, 鄭 正 和
漢陽大學校 工科大學 電子工學科
CAD 및 통신회로연구실

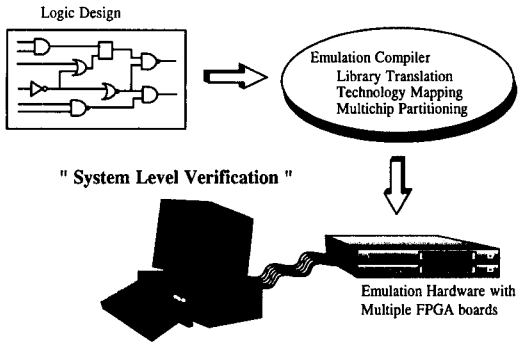
I. 서 론

반도체 설계 기술이 발전함에 따라 1980년대 후반에 FPGA가 출현함으로서 기존의 주문형 반도체인 게이트 어레이나 스탠다드 셀에 의존해 왔던 ASIC 시장에 큰 변화가 일어나고 있다. 모든 전자제품이 소형, 경량화가 되어가고 있으며 제품의 개발 주기가 짧아짐에 따라 회로 부품의 변화가 빠르게 이루어지고 있는 상황속에서 특히 FPGA가 갖는 장점인 재프로그램 가능성과 end-user에 의한 신속한 회로 설계로 많은 각광을 받고 있어 ASIC 시장에서 주목받는 소자로 부각되고 있다.

근래에 들어서는 ASIC으로 구현되는 회로의 크기가 100만 게이트 급에 이르고 있고 이를 하나의 칩에 구현하고자 하는 추세에 따라 ASIC의 검증 문제가 중요한 과제로 대두되었다. 보다 빠르고 정확한 검증은 일각을 다투는 전자 산업의 시장에서 보다 상대적으로 신속하게 제품을 생산할 수 있고, 제품 생산까지의 시행착오를 줄여줄 수 있기 때문에 상당히 중요한 의미를 갖는다.

ASIC을 검증하기 위해서 종래에는 주로 소프트웨어적인 시뮬레이션에 의존하여 왔으나, 소프트웨어적인 시뮬레이션 접근 방식은 실제 상황이 아닌 가상적인 모델링에 의한 것이기 때문에 신뢰성에 한계가 있으며 더욱이 근래에는 시뮬레이션 소프트웨어와 컴퓨터의 성능이, 급속도로 증가하는 회로의 크기 및 복잡도를 따라가지 못하고 있다. 따라서 최근 이러한 시뮬레이션의 단점을 극복하고 보다 정확한 검증을 위하여 실제 상황에 가깝고, 빠른 검증을 수행할 수 있는 하드웨어적 에뮬레이션 기법에 많은 연구가 집중되고 있다.

이러한 하드웨어 에뮬레이션은 종래에는 설계한 회로와 기능적으로 동등한 시제품을 CAD 소프트웨어와 PCB 설계 기술을 이용하여 직접 제작, 검증하였으나 시제품 제작에 많은 시간과 비용이 소요되고 설계상의 오류가 있거나 회로가 변경될 경우 반복해서 시제품을 제작해야 하므로 재설계 시간의 부담과 설계 비용이 추가되는 문제가 생긴다.



〈그림 1〉 ASIC 에뮬레이션 시스템을 이용한 회로 검증

FPAG를 이용한 검증은 FPGA의 성능 및 구현 가능한 회로의 규모 등에 의해 1개의 FPGA를 이용하여 구현할 수 없는 대규모의 회로인 경우 필요할 때마다 여러 개의 FPGA를 연결하기 위해 PCB를 제작해야 하는 문제가 생기므로 이러한 문제점을 해결하고자 FPGA간의 결선이 이미 이루어져 있는 범용성 FPGA 즉 ASIC 에뮬레이터의 개념이 출현한 것이다. ASIC 에뮬레이터를 이용하면 설계된 회로를 실제적으로 FPGA에 구현하여 검증할 수 있으며 또한 회로의 변경에 따른 새로운 PCB를 제작할 필요가 없어진다.

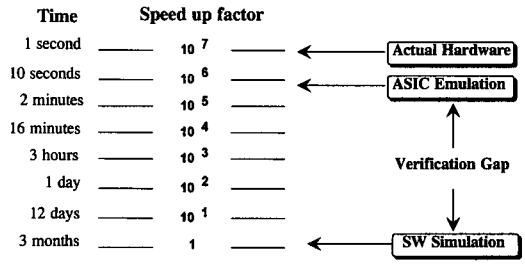
본 고에서는 여러 개의 FPGA 칩을 어떤 특정한 배선 구조로 연결한 후, 재프로그램 가능한 에뮬레이션 보드에 설계된 회로를 자동으로 구현하고 기능을 신속하게 검증할 수 있는 에뮬레이션 시스템에 대하여 기술한다.

제 II 장에서는 ASIC 에뮬레이션 시스템 설계에 대한 일반적인 사항에 대하여 논의하고, 제 III 장에서는 기존에 발표 또는 상품화된 ASIC 에뮬레이션 시스템에 대하여 고찰하고, 제 IV 장에서는 에뮬레이션 시스템의 설계 사례를 구체적으로 설명하고 제 V 장에서 결론을 맺는다.

II. 에뮬레이터의 기본개념

요즘의 ASIC 개발 추세는 가능하면 많은 기능을 가지면서 소형화함으로서 다른 제품보다 기술과 성능면에서 우월성을 갖도록 하여 제품 경쟁력에서 선두를 유지하기 위해 치열한 연구를 하고 있다. 90년대 초에 이르러 백만 게이트급이상의 회로를 하나의 칩에 집적화시킬정도로 집적 회로 설계 기술은 나날이 발전해 가고 있다. 그러나 이러한 집적회로 설계에 있어 가장 큰 문제가 되는 것은 회로의 기능에 관한 검증 문제이며 검증을 할 수 있다하더라도 어떻게 하면 실제 상황과 유사한 환경속에서 실시간에 가깝게 검증하는가이다.

대표적인 예가 그림 2에 나타나 있다. 이 자료는 10만 게이트의 ASIC 회로를 검증하는데 걸리는 시간을 비교한 것이다.



〈그림 2〉 검증 방식에 따른 속도 비교

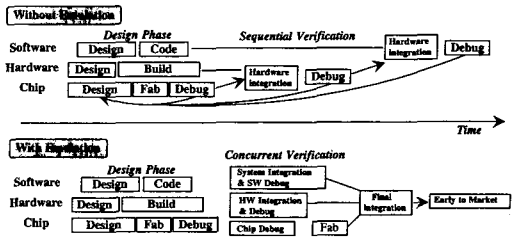
즉, 실제 시스템을 하드웨어로 설계하여 1초 동안에 검증할 수 있는 문제를 워크스테이션에서 소프트웨어적으로 시뮬레이션 할 경우에는 3개월이상을 소요 그대로 하면서 풀 수 있다는 것이다. 이런 경우에 에뮬레이터를 사용하면 시뮬레이션에 비해 백만배 이상의 검증 속도를 향상시킬 수 있다는 것을 이 그림으로 부터 알 수 있다. 또 한가지의 문제는 소프트웨어적인 시뮬레이션 결과에 대한 신뢰성에 대한 문제이다. 방대한 양의 테스트 벡터 자료를 입력으로 하여 소프트웨어적으로 모델링한 회로에 대한 결과이므로 실제적이지 못하다는 점이다. 결국 하드웨어로 구현될 시스템은 하드웨어로 모델링하여 입력 자료 역시 실제 시스템에서 사용하는 전기적인 신호를 인가하여 정상적으로 동작하는지의 여부를 검증하는 것이 에뮬레이션의 기본 개념인 것이다.

ASIC 에뮬레이터는 ASIC이 필요한 모든 현장에서 반도체 회사에 개발을 의뢰하지 않고 모든

현장에서 직접 사용할 수 있는(off-the-shelf) 칩을 개발하고자 하는데 목적이 있다. 기존의 장시간을 요하는 시뮬레이션 기법 및 방대한 테스트 벡터로 인하여 검증이 불가능한 설계를 단시간 내에 검증이 가능하게 하는 에뮬레이션 시스템이야말로 ASIC 설계에 있어 필수 불가결한 요소라 할 수 있다. 결과적으로 에뮬레이션 시스템을 사용함으로써 목적 회로를 빠르고 정확하게 구현, 검증할 수 있으므로 최종적으로 원하는 제품의 빠른 시장 진출을 유도할 수 있으며, 재설계로 발생할 수 있는 비용, 시간 및 노력을 최소화할 수 있다.

에뮬레이션의 잇점으로는 소프트웨어와 하드웨어 설계를 동시에 진행할 수 있고 시제품과 거의 동등한 하드웨어를 빠른 시간내에 획득할 수 있다는 점과 재사용 가능한 소자를 사용함으로써 개발 비용을 줄일수 있다. 에뮬레이션 시스템은 소프트웨어적으로는 확인할 수 없는 실제 데이터의 흐름을 확인할 수 있다는 장점이 있으나 FPGA가 갖는 저속의 동작 속도로 인하여 일정 규모 이상의 회로에 대해서는 실시간 처리가 불가능하고 메인 클럭의 주파수를 낮추어 회로를 동작시키면서 검증을 실행해야 한다. 따라서 실제 시스템과 실시간으로 동작시켜야만 검증할 수 있는 경우엔 곤란한 문제가 생긴다. 이런 경우에는 주변회로의 동작 속도를 낮추어 실행하거나 개발될 ASIC의 기본적인 기능 정도만 확인하는 것으로 해결할 수 있다. 현재 제품화되고 있는 에뮬레이터의 가격이 엄청나게 고가인 점도 에뮬레이터가 ASIC 설계에 보편화되어 사용되고 있지 않은 이유 중에 하나이다. 또 다른 단점은 사용 방법이 너무 복잡하고 어려워서 회로의 크기가 매우 큰 경우엔 검증 시간의 대부분이 에뮬레이터의 복잡한 조작이 차지한다는 점이다.

일반적으로 ASIC 에뮬레이션을 수행하는 시점은 논리 설계가 종료되고 설계가 검증된 결과인 게이트 레벨의 회로로 부터 시작된다. ASIC 에뮬레이터를 사용할 때의 설계 과정과 사용하지 않을 때의 설계 및 검증 과정을 비교하면 그림 3과 같다.

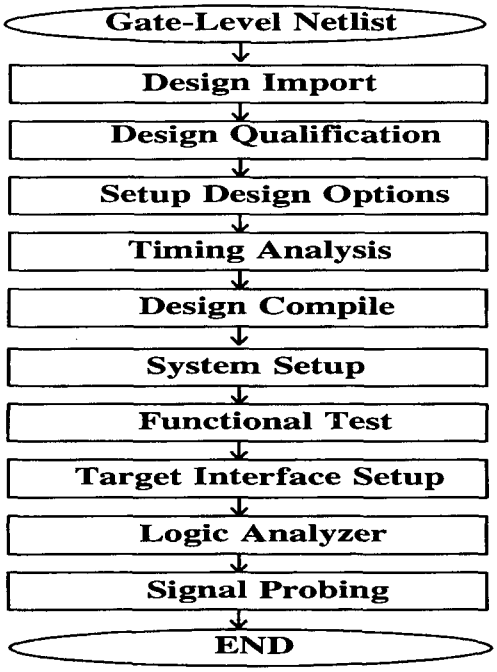


〈그림 3〉 설계 검증 과정의 비교

에뮬레이터를 사용하지 않을 경우에는 게이트 레벨의 회로도로 부터 배치와 배선을 수행하고 레이아웃 검증을 수행한 후 마스크 패턴으로 부터 칩을 제작한다. 제작된 칩을 검증한 후 설계된 모든 하드웨어와 결합하여 정상적으로 동작하여 설계 오류가 없을 경우에 다시 설계된 소프트웨어와 맞물려 전체 시스템을 동작시키면서 시스템 검증을 수행한다. 이와 같은 과정을 시스템이 정상적으로 동작할 때까지 재설계를 반복수행한다. 즉, 하드웨어와 소프트웨어를 동시에 병렬적으로 검증을 수행할 수 없기 때문에 자연히 개발기간이 늘어나게 된다.

그러나 ASIC 에뮬레이터를 사용하게 되면 그림에서 보는 바와 같이 칩을 제작하기 이전에 칩을 비롯한 하드웨어 설계와 소프트웨어 설계를 결합하여 동시에 검증(concurrent verification)을 수행하고 나서 설계 오류가 없다는 것이 확인된 후 칩을 제작함으로서 재설계로 인한 설계 시간 및 설계 비용을 단축시킬 수 있고 빠른 시장 진출로 인하여 제품 경쟁력을 제고시킬 수 있다.

ASIC 에뮬레이션 과정은 그림 4와 같은 과정으로 진행된다. 논리 설계 과정을 마친 게이트 레벨의 연결정보를 읽어 들여 데이터베이스를 구성한다. Design qualification 단계에서는 읽어들인 연결 정보에 대하여 오류가 있는지 점검을 하고 전체 게이트의 개수, 구현가능한 보드의 개수 및 FPGA 칩의 갯수, 기본 입출력의 개수등 입력 회로에 대한 여러 가지 정보를 추출한다. 다음 단계에서는 검증을 원하는 신호선의 리스트, critical net, 초기 분할 방법등 에뮬레이션 컴파일을 위한 옵션들을 설정한다. Timing analyzer는 입력 회로



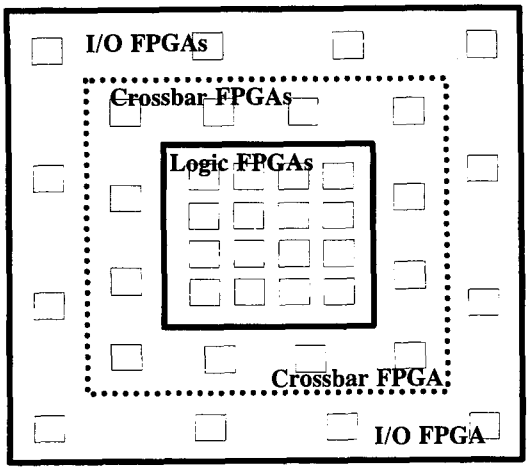
〈그림 4〉 AISC 에뮬레이션 과정

에 대하여 각 소자의 지연 시간을 고려하여 각 신호선에 대한 도착 시간, 출력에 나타나야 하는 시간 등의 정보를 계산하여 파일로 출력한다.

ASIC 에뮬레이션 소프트웨어의 핵심 부분인 design compile 단계에서는 입력 회로에서 사용한 셀 라이브러리를 에뮬레이션 시스템에서 사용하는 FPGA 라이브러리로 변환시키는 라이브러리 변환(library translation), 회로의 계층 구조를 모두 펼친후 FPGA가 사용하는 기본 논리 블록의 구조에 알맞고 시간 제약 조건을 위배하지 않게 할당하는 기술 매핑(technology mapping), 목적 회로의 크기가 커서 하나의 FPGA 칩에 구현이 불가능 할 경우 타이밍 정보와 기술 매핑 결과인 기본 논리 블록 레벨의 연결 정보를 이용하여 여러 FPGA 칩에 주어진 회로를 분할하는 회로 다중 분할, 각각의 FPGA 칩별로 기본 논리 블록의 배치와 이들사이의 연결 관계를 고려한 배선, FPGA 칩에 대한 프로그램을 위한 출력 파일의 생성 등의 과정을 수행한다. 마지막으로 프로그램 파일을 FPGA 보드에 다운로드(download)한다. FPGA 보드의 논리 구현용 FPGA에는 프로그램이 완료되었으므로 논리구현용 FPGA 간의 연결 관계를 만족시키도록 배선전용 FPGA에 프로그램하여 다운로드한다. FPGA 칩의 프로그래밍을 마친 후에는 FPGA 보드와 주변 회로와의 접속을 마친후 전체 회로를 동작시키면서 기능을 점검한다. 회로의 동작에 오류가 발생할 경우 논리분석기(logic anlyzer)를 이용하여 검증을 원하는 신호선을 관찰하여 설계 오류를 찾고 재설계한 후 위의 과정을 반복한다.

ASIC 에뮬레이터는 크게 FPGA칩과 이들사이의 연결 구조, 논리 분석기 등의 에뮬레이션 하드웨어와 주어진 회로의 연결 정보로부터 자동으로 FPGA 칩에 ASIC을 구현해 주는 에뮬레이션 소프트웨어들로 구성된다. 에뮬레이션 하드웨어의 코아부분은 FPGA 보드로서 FPGA 칩을 사용하여 PCB로 설계하는 것인데 FPGA의 용도에 따라 그림5와 같이 크게 3가지로 나눌 수 있다. 첫째, 논리 구현용 FPGA로 사용자가 설계한 ASIC 회로가 프로그램되는 FPGA를 말한다. 둘째, 신호선 연결용 FPGA로 논리 구현용 FPGA 사이의 연결 또는 논리 구현용 FPGA와 입출력 용 FPGA의 연결을 하기 위한 FPGA로 crossbar라고도 부른다. 최근에는 배선만을 담당하는 연결전용칩을 사용하기도 한다. 셋째, 입출력용 FPGA로 이것은 ASIC 에뮬레이터로 구현된 시스템의 시제품을 검증 또는 입출력을 위해 외부 시스템과의 연결을 사용자가 임의로 설정할 수 있도록 제공되는 FPGA를 말한다.

에뮬레이션 보드를 설계할 때 고려해야 할 사항은 먼저 목적하는 에뮬레이션 시스템이 수용할 수 있는 게이트 용량이다. 이에 따라 사용할 논리구현용 FPGA 칩의 종류와 개수가 선정된다. 여러 FPGA마다 각각 특성이 다르므로 칩의 지연 시간, 칩당 게이트 이용율(최대, 평균, 게이트의 기준 등을 포함), 재프로그램성, 칩을 구성하는 기본 논리 블록의 구조 및 이들 사이의 배선 구조, 시스템 내에서의 프로그램 가능성 등을 고려해야 한다. 이때 유의할 점은 제조회사의 자료를 100% 믿지 말고 가능하면 그 칩을 사용해 본 유경험자의 조언을 참고하는 것이 바람직하다. 논리구현용 FPGA 칩



〈그림 5〉 ASIC 에뮬레이터 FPGA 보드의 일반적인 구조

이 선정되었으면 이들 사이의 연결 구조에 대한 설계를 진행해야 한다. 연결 구조에 따라 FPGA 사이에 연결 가능한 신호선의 개수에 대한 제약 조건이 달라지기 때문이다. FPGA의 특성상 칩 내부의 지연 시간은 거의 일정하나 칩 외부와의 지연 시간이 회로의 동작에 결정적인 영향을 미치므로 회로의 분할 과정에서는 물론 FPGA 칩간 연결 구조의 설계시 지연 시간의 최소화와 균등화를 달성할 수 있는 효과적인 연결 구조가 필요하다. 선정된 FPGA와 더불어 지원되는 라이브러리의 종류, CAD 소프트웨어에 대하여 신중히 검토하여야만 한다. 설계된 ASIC 회로의 게이트 레벨 회로에 대한 출력 형태와 FGPA 설계용 tool과의 인터페이스가 용이한가를 점검하는 것은 당연하다. 또한 design compile의 중간 과정에서 다른 업체의 CAD tool을 사용할수 있는 입출력 조건을 만족하는지 파악해야만 사용자가 개발한 tool을 컴파일 중간 과정에 사용할 수 있고 벤취마크 테스팅이 가능하다.

ASIC 에뮬레이터를 지원하는 CAD 소프트웨어에는 위에서 언급한 라이브러리 변환기, 타이밍 분석기,기술 매퍼,다중 회로 분할기, 배치배선기외에 논리분석기와 인터페이스하면서 검증 신호선에 대한 정보를 관리하면서 파형을 디스플레이하는 보조프로그램과 design compiler를 직접 개발할 경우에는 출력화일을 생성하는 비트스트림 데이터 파일 생성프로그램이 따로 필요하다.

대부분의 전자회로 시스템은 마이크로프로세서나 DSP에 의해서 수행되는 소프트웨어 부분과 ASIC에 의해 수행되는 하드웨어 부분으로 나뉘어져 설계가 진행된다. 따라서 최근 고밀도 집적회로 설계시 성능과 비용을 고려하여 효과적으로 설계하기 위하여 하드웨어와 소프트웨어가 혼합된 시스템을 체계적으로 분할, 시뮬레이션, 합성, 인터페이스하여 설계하는 하드웨어/소프트웨어 통합설계에 대한 연구가 활발히 진행되고 있다. 특히 통합 시스템 설계에서 중요한 것은 어떤 부분을 하드웨어로 설계하고 어떤 부분을 소프트웨어로 구현할 것인가를 결정해주는 문제이다. 이때 하드웨어와 소프트웨어의 접속 부분에서 지연 시간이 최소가 되도록 설계해야 하드웨어의 성능 향상 효과를 제대로 얻을 수 있다. 최근의 ASIC 에뮬레이터에서도 하드웨어/소프트웨어 통합설계 개념을 도입하여 검증하는 추세에 있다. 설계하고자 하는 전체 시스템을 하드웨어와 소프트웨어로 분할하여 소프트웨어로 구현하는 것이 비용함수가 적은 부분은 마이크로프로세서나 DSP에 프로그램하여 설계하고, 하드웨어로 설계하는 것이 효율적인 ASIC 부분은 FPGA 보드에 실현한 후 상호 인터페이스하면서 전체 시스템을 검증한다. 이러한 에뮬레이터를 MCU/DSP embedded ASIC 에뮬레이터라 하고 DSP의 코아 부분, 마이크로콘트롤러, RAM, ROM 그리고 그밖의 주변회로가 에뮬레이션 하드웨어에 추가된다.

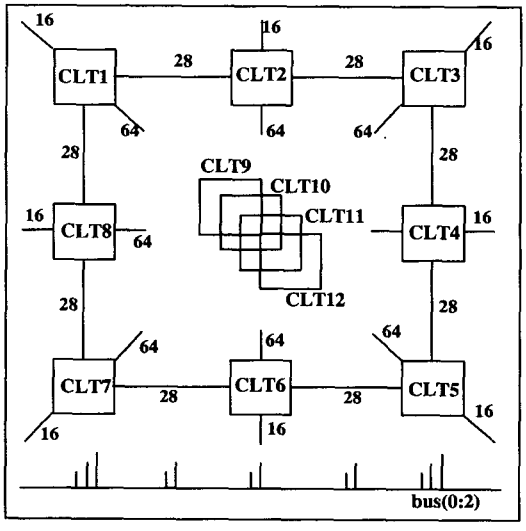
III. 기존의 에뮬레이터

ASIC 에뮬레이션 시스템은 1990년대 초 ASIC이나 주문형 IC를 제작하기 전에 전체적인 시스템의 시제품을 구현하고자 하는 목적으로 상품화되기 시작하여 InCA, PIE, Aptix, Quickturn등의 여러 회사에서 에뮬레이터를 상품화하여 왔고 최근 Quickturnt사에서는 UltraSPARC-I의 에뮬레이

션을 수행한 결과를 발표한바 있다. 상품화된 ASIC 에뮬레이터는 에뮬레이션 하드웨어와 더불어 주어진 회로의 자동분할, 기술매핑, 배치, 배선 기능 및 구현된 설계 회로를 디버깅할 수 있는 여러가지 기능이 제공되고 있다. 또한 디지탈 시스템에서 많이 사용되는 마이크로프로세서, 메모리, DSP등이 포함되어 설계된 ASIC도 에뮬레이션 시스템에 구현하여 쉽게 검증할수 있는 제품을 선보이고 있다.

1. InCA의 에뮬레이터

InCA의 ASIC에뮬레이터는 VA(Virtual ASIC)와 CS(Concept Silicon)로 나누어 진다. CS는 에뮬레이션 보드에 설계된 회로를 구현하기 위해 개발된 에뮬레이션 컴파일러를 말한다. VA는 CAT6와 CAT12라는 두 개의 에뮬레이션 하드웨어 모듈을 갖는다. CAT6 에뮬레이션 모듈은 6개의 XC3090으로 구성되고 CAT12 에뮬레이션 모듈은 12개의 XC3090으로 구성되며 그림 6과 같은 구조로 되어 있다.



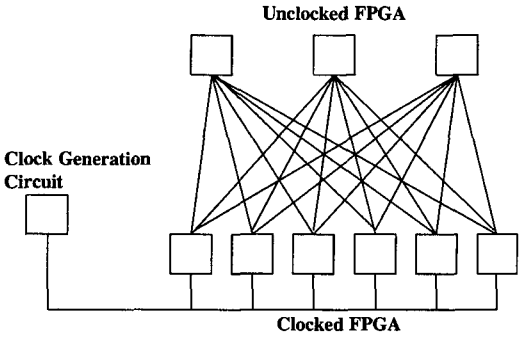
〈그림 6〉 CAT12-128의 내부 구조

그림 6에서 내부에 있는 CLT9에서 CLT12까지의 FPGA는 crossbar의 역할을 한다. 또한 CLT1에서 8까지는 논리 회로를 구현하는 FPGA이다. 이 구조의 특징은 외부 시스템과 연결되는 I/O 신호를 crossbar를 이용하지 않고 직접 논리를 구현하는 FPGA로 연결하고 논리를 구현하는 FPGA 중 인접된 FPGA 간에는 직접적인 연결이 가능하도록하여 crossbar를 통과함으로서 생기는 전달 지연 시간을 단축하려고 노력하였다. VA는 에뮬레이션 모듈을 8개까지 확장할 수 있기 때문에 에뮬레이션 모듈간의 연결은 외부 포트를 이용하여 연결한다. 이러한 구조에 의해 InCA에서는 20MHz까지 에뮬레이션이 가능하다고 발표하고 있다. 또한 InCA에서 VA-II를 발표하였는데 VA-II의 특징은 기본적으로 3만 게이트를 구현할 수 있는mother 보드와 DSP, 마이크로프로세서, RAM, ROM 등을 각각 탑재하는 daughter 보드의 개념을 사용하는 확장성에 있다. Daughter 보드는 별도의 연결 장비없이 직접 mother보드에 연결 가능하며 보드당 512개의 입출력 채널을 가진다. FPGA는 XC3042, XC3090, XC4010을 사용하고 에뮬레이션 속도는 20MH정도이며 VA의 단점인 probing 기능을 추가하여 probing 핀을 지원하는 84핀 XC3042를 사용한다.

2. PIE의 에뮬레이터

PIE의 ASIC 에뮬레이터인 MARS-II는 MP(Modular Packaging)와 IP(Integrated Package)로 구분되며 MP는 LBM(Logic Block Module)에서 구현할 수 있는 게이트의 크기에 따라 세 종류로 구분된다. 즉, 25K 게이트를 구현할 수 있는 LBM2, 50K 게이트를 구현할 수 있는 LBM3-S와 100K 게이트를 구현할 수 있는 LBM3-D로 나뉜다. 설계한 회로의 크기가 증가하면 InCA사의 ASIC 에뮬레이터는 슬럿 확장 개념으로 에뮬레이션 모듈을 추가시키지만 PIE사의 ASIC 에뮬레이터는 LBM을 교체하여야 한다. 만일 LBM을 교체하지 않고 몇 개의 LBM으로 설계한 회로를 구현하여 에뮬레이션 하려면 커넥터를 통하여 연결해야 한다. MARS-II의 특징은 임의의 외부 포트에서 회로 내부의 신호를 probing할 수 있다는 점이다. VA-II(InCA)의 경우는 84핀 XC3042를 이용하여 제한된 pin의 probing을 할 수 있지만 MARS-II(PIE)의 경우는 하나의

LBM에서 1000핀 정도의 probing이 가능하다. 또한 논리 회로를 구현하는 회로 중 clock을 발생시키는 회로들을 클럭 경로의 분석 후 하나의 클럭 전용 FPGA로 할당한다. 한편 Xilinx사의 X4000 계열이 갖는 메모리 모델링 기능을 이용하여 메모리의 모델링이 가능하다. LBM2의 경우 FPGA는 Xilinx사의 XC4005를 사용하며 그 구성은 다음과 같이 예측된다. 28개 논리 구현용 FPGA는 38개의 crossbar와 I/O용 FPGA으로 구성되어 있는 것으로 추정된다. MARS-II의 LBM 내부 구조를 보면 그림 7과 같은 구조로 되어 있다.

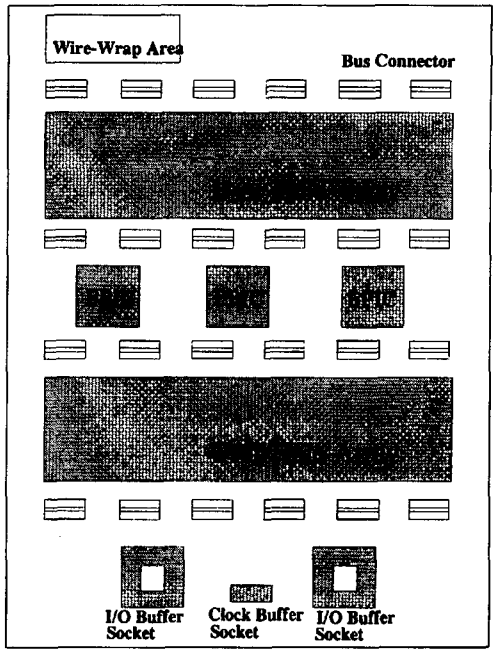


〈그림 7〉 MARS-II의 내부 구성

그림 7에서 보면 clocked FPGA는 논리를 구현하기 위한 FPGA이고 unclocked FPGA는 crossbar 역할을 담당하는 FPGA이다. Unclocked FPGA가 crossbar로 사용되지 않을 때는 논리 구현용 FPGA 역할을 하게 된다.

3. Aptix의 에뮬레이터

Aptix사에서는 다층 FPCB(Field Programmable Circuit Board)상에서 배선전용 칩인 FPIC(Field Programmable Interconnect Component)를 이용한 에뮬레이터를 상품화하고 있다. FPIC는 1024개의 핀을 갖는 연결전용 칩으로 936개를 일반 신호선의 연결에 사용할 수 있고 64개의 핀을 외부의 로직 어날라이져와 연결하여 검증을 원하는 신호선을 관측하면서 디버깅할 수 있다. FPIC는 임의의 두 핀간 지연 시간이 10ns 정도의 고속연결용 프로그램가능 칩이다. Aptix사



〈그림 8〉 MP3 보드의 구조

에서는 Explorer series로 에뮬레이터를 개발하였는데 Explorer ASIC, Explorer MP3, Explore MP4가 대표적인 예이다.

Explorer ASIC은 Xilinx사의 XC4000계열의 FPGA를 위한 21개의 소켙과 4개의 FPIC, 2개의 EPROM 소켙, 그리고 wire-wrap 영역으로 구성된다. 21개의 소켙중 16개는 XC4010, XC4008, XC4003H를 이용하여 논리를 구현하고, 입출력 버퍼와의 연결을 위하여 4개의 XC4005칩을 사용하고, 나머지 하나는 고속의 클럭트리 구성을 위해 XC4003칩을 사용한다. Explorer ASIC을 사용하여 약 20만 게이트의 회로를 구현할 수 있고, 외부의 로직 어날라이져를 이용하여 384개의 회로 내외부 신호를 동시에 probing할 수 있다.

Explorer MP3의 에뮬레이션 보드는 그림8과 같이 FPGA/CPLD, DSP, RAM/ROM, ASIC 및 표준소자들을 탑재하기 위한 1920개의 user hole array가 추가된 것이 특징이며, 3개의 FPIC, 클럭 버퍼 소켙, 그리고 32x12 wire-wrap 영역, 입출력 버퍼 소켙, 오실레이터 커넥터, FPGA download

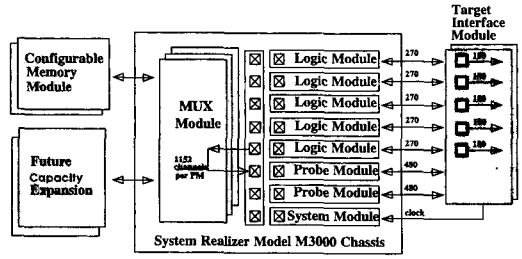
커넥터등으로 구성된다. MP3 보드는 12층으로 8층 PCB로 제작되어 이중 8개의 층을 신호선 연결을 위해 사용한다. 3개의 FPIC 사이에는 100개의 신호선이 완전그래프 형태로 연결되어 있다. 외부의 로직 어날라이져를 이용하여 192개의 신호선을 동시에 관측하고 디버깅할 수 있다.

Explorer MP4는 2880개의 free hole과 608개의 입출력 신호선으로 구성되는 에뮬레이션 보드와 4개의 FPIC로 구성되는데 에뮬레이션 보드에는 Xilinx XC4000 칩을 약 20개가량 탑재할 수 있다. 특히 MP4는 고속의 80비트 폭의 버스 연결을 지원한다.

4. Quickturn의 에뮬레이터

Quickturn사에서는 Logic Animator, MARSⅢ, System Realizer등의 에뮬레이터를 상품화하여 제품으로 내놓고 있다. Logic Animator의 용량은 약 50,000게이트 정도로 큰 용량은 아니지만 에뮬레이션 속도는 8MHz에서 16MHz정도로 비교적 빠른 편이다. 에뮬레이터에 구현된 회로의 내외부 신호선을 타사의 로직 어날라이저를 이용해서 448개까지 동시에 porbing 할 수 있다. 특히 Logic Animator는 비교적 크지 않은 회로를 목표로 하여 주로 클럭 신호선과 최장경로의 타이밍을 고려하여 설계했으므로 실제 시스템의 동작 속도와 거의 유사하게 시스템을 에뮬레이션할 수 있다. MARSⅢ는 Quickturn이 인수한 PIE의 MARSⅡ를 개선한 것으로 Xilinx의 XC4005, XC3090을사용했으며 기본적으로 10만게이트까지 지원할 수 있으며 선택에 따라 백만 게이트까지 구현 가능하다. 로직 어날라이저를 내장하고 있어서 회로의 내외부 신호를 분석할 수 있으며 1024개의 신호선을 동시에 검증할 수 있고 2000개의 트리거상태를 가질수 있다. ASIC 내부에 있는 작은 용량의 ROM, RAM은 FPGA에 컴파일시키고 큰 용량은 외부 보드로 지원되게 만들어져 있다. 비디오 신호와 같이 빠른 신호가 에뮬레이션으로 늦게 처리되는 경우는 처리된 데이타를 모아서 한꺼번에 디스플레이하는 버퍼링 기능을 갖고 있다.

System Realizer는 M250이 기본 모델인데



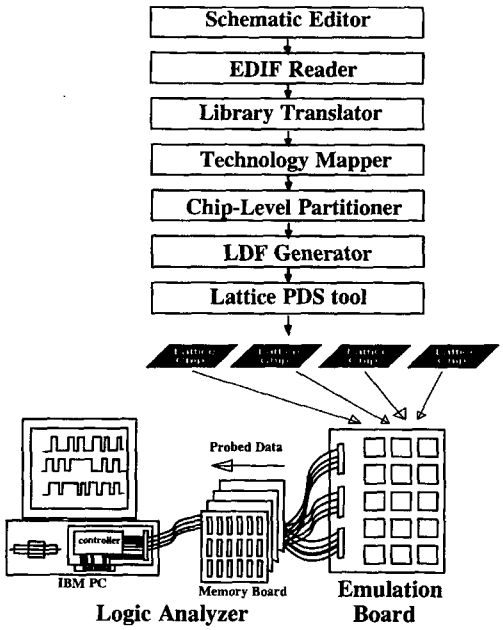
〈그림 9〉 Quickturn의 M3000의 구조

Xilinx의 XC4013을 사용해서 250,000게이트까지 에뮬레이션할 수 있으며 M250 모듈들을 추가하여 M3000모델을 만들 수 있는데, M3000은 3백만 게이트까지 지원할 수 있다. 에뮬레이션 속도는 M3000은 4MHz이하이고 M250은 4MHz에서 8MHz정도로 발표되고 있다. 그림 9는 M3000의 내부 구조를 나타낸다.

M3000은 두 단계의 backplane구조로 설계되어 확장성이 용이하고 각 모듈을 독립적으로 컴파일할 수 있다. 55개 ASIC 라이브러리를 사용하여 Xilinx라이브러리로 변환시킬 수 있다. Logic module은 25만 게이트까지 구현 가능하며 메모리를 포함하는 회로도 컴파일 가능하다. M3000의 입출력 신호선의 수는 700개부터 14000개까지 확장 가능하다. 로직 어날라이저 기능을 내장하고 있으며 각 probe module은 12개의 bank로 구성되는데 각 bank는 128KB의 depth를 갖는 96개의 채널로 총 1152개 신호를 동시에 검증할 수 있다.

Ⅳ. 에뮬레이터의 개발 사례

본 장에서는 한양대학교 CAD 및 통신회로 연구실에서 직접 자체 개발한 ASIC 에뮬레이터를 예를 들어서 개발사례를 기술한다. 본 연구실에서는 두 가지 종류의 에뮬레이터를 개발하였는데 하나는 FPGA 칩만으로 구성된 ASIC 에뮬레이션 시스템이며 다른 하나는 MPU와 DSP가 내장된 에뮬레이션 시스템이다. 에뮬레이터(I)의 전체 시스템 구성은 그림 10과 같다. 개발된 에뮬레이션 시스템



〈그림 10〉 개발된 에뮬레이션 시스템(I)

은 FPGA 보드, 논리분석기로 구성되는 에뮬레이션 하드웨어와 EDIF 번역기, 라이브러리 변환기, 기술 맵퍼, 칩단위 회로분할기, LDF 생성기로 구성되는 에뮬레이션 소프트웨어로 이루어진다.

FPGA 칩은 Lattice사의 ispLSI1048 FPGA 칩을 사용하였다. Lattice사의 FPGA 칩으로 선정한 이유는 개발 당시 동작 속도가 빠르고, 입출력 핀간의 지연 시간이 짧은 장점이 있었기 때문이다. 이외에도 재프로그램과 시스템 내에서의 프로그램이 가능하고, 배선 자원이 풍부하여 기본 논리 블럭간의 연결을 위한 배선 문제를 크게 고려하지 않아도 된다는 장점을 갖고 있다.

FPGA 보드는 Lattice사의 ispLSI1048 FPGA 칩 15개와 40개의 핀을 갖는 입출력 포트 18개로 구성된다. 사용된 ispLSI1048칩은 최대 8K 게이트의 회로를 구현할 수 있고 전체 120개의 핀 중에서 96개를 일반적인 입출력 핀으로 이용할 수 있다. 에뮬레이션 보드에서 사용한 15개의 칩 중에서 12개를 논리구현용으로 사용하고 3개를 배선 전용 칩으로 사용하였다. 검증 가능한 회로의 크기는 이론적으로 최대 96K게이트이지만, 회로를 FPGA에 구현할 때 배치 및 배선의 어려움 때문에 통상적으로 게이트 이용률을 약 30%에서 40% 정도로 계산하여 실제 검증 가능한 회로의 크기는 30K게이트에서 40K게이트 정도이다. 외부 시스템과의 연결을 위해 총 288개의 입출력 핀을 가지고 있으며 내부 신호의 검증을 위해 최소 96개의 핀을 가진다.

논리 구현용 칩 12개를 세 개의 그룹으로 나누어 각 그룹당 4개씩의 칩을 완전 그래프 구조로 연결하고, 그룹간의 연결을 위하여 나머지 3개의 칩을 배선 전용 칩으로 사용한다. 배선 전용 칩은 모든 논리 구현용 칩의 핀들과 연결되어 있어서 그룹간의 배선을 프로그램할 수 있으며 배선 전용 칩 사이에는 연결 관계가 없다.

ASIC 에뮬레이터로 구현된 회로가 정상적으로 동작하는지 검증하기 위해서는 원하는 신호선의 파형을 관측하고 분석하는 논리분석기(logic analyzer)가 필수적이다. 논리분석기는 검증을 원하는 지정된 신호선에 대하여 회로 동작 중의 상태 변화를 일정한 샘플링 간격으로 저장한 후 지정된 신호선의 파형을 그래픽으로 출력한다. 논리분석기의 하드웨어는 데이타를 저장할 SRAM 및 부가적인 논리 회로로 이루어져 있는 메모리 보드와 메모리 보드들을 제어하고 PC와의 인터페이스를 담당하는 인터페이스 보드로 구성된다. 메모리 보드는 하나의 보드에 여러 개의 SRAM을 탑재시킬 수 있는데 현재는 4개의 SRAM으로 구성되어 있다. 각 SRAM은 8개의 채널로 사용되므로 모두 32채널을 갖게 된다. 이러한 메모리 보드를 64개까지 연결할 수 있도록 설계되어 있다. 신호 파형의 샘플링은 사용자가 지정한 일정한 샘플링 간격에 따라서 주기적으로 신호의 샘플을 취해서 SRAM에 저장한다. SRAM에 데이타를 모두 저장해서 메모리가 차게 되면 PC에 하드웨어 인터럽트 신호를 발생하여 데이타의 샘플링이 종료되었음을 알리고 SRAM에 저장된 데이타를 디스크에 화일의 형태로 저장한다. SRAM에 저장된 데이타를 화일로 저장할 때에는 메모리 보드의 SRAM을 PC가 직접 접근해서 데이타를 읽는다. 논리 분석기가 다루어야 하는 전체 데이타의 수는 매우 많

으므로 어드레스 공간도 메가 바이트 단위를 필요로 한다. 그러나, PC에서 사용할 수 있는 어드레스의 빈 공간은 제한되어 있으므로 SRAM의 전체 주소 공간을 여러 개의 뱅크로 나누어 사용한다. 각 SRAM의 뱅크로부터 샘플링 데이터를 읽어 들일 때마다 PC상에 어드레스 맵핑을 하여 각 뱅크로부터 차례로 읽어오게 된다. 데이타가 모두 디스크에 저장되면 사용자의 요구에 따라서 데이타를 모니터 상에 그래픽 파형으로 출력하도록 프로그램되어있다.

설계자가 회로 편집기상에서 작성하거나 VHDL로부터 자동 생성된 회로를 EDIF의 형태로 출력하면 EDIF 번역기는 회로를 기술하는 EDIF 형식의 화일에 대하여 LEX와 YACC를 이용하여 구문 분석(syntax analysis)과 의미 분석(semantic analysis)을 하여 전체 회로를 구성하는 회로 소자간의 연결 정보(netlist)를 추출한다. 또한 개발한 EDIF 번역기는 계층적(hierarchical)으로 기술된 EDIF 입력 화일도 처리가 가능하도록 설계되어 있다.

EDIF 번역기로부터 출력된 연결 정보는 특정 회사에서 개발된 ASIC 라이브러리로 기술되어 있고, ACE의 에뮬레이션 보드는 Lattice사의 FPGA 칩으로 구성되어 있으므로 주어진 회로를 에뮬레이션 보드상에 구현하기 위해서는 Lattice사에서 제공하는 라이브러리들간의 연결 정보로 변환시켜야 하는데 이와 같은 역할을 라이브러리 변환기가 수행한다. 회로를 기술하는 라이브러리의 종류와 형식은 매우 다양하므로 라이브러리 변환기의 입력 요구 조건에 맞는 타이밍을 고려한 입출력 포트 맵핑 정보를 입력 화일의 형태로 기술해 주어야 한다. ACE는 어떠한 종류의 라이브러리라도 라이브러리 변환기의 입력 기술 형식에 따라 입력해 주면 Lattice사의 FPGA에서 적용 가능한 형태의 연결 정보로 출력해 준다.

기술 맵퍼는 라이브러리 변환기로부터 출력된 회로의 연결 정보로부터 Lattice사의 FPGA 칩을 구성하는 기본 논리 블럭인 GLB 단위로 주어진 회로를 분할하여 각각의 기본 논리 블록의 구조에 알맞게 할당하고 칩 단위 회로 분할기에 기본 논리 블럭간의 연결 정보를 출력한다. 기술 맵퍼의 목적 함수는 전체 기본 논리 블럭 개수의 최소화와 기본 입력으로부터 기본 출력까지의 기본 논리 블럭 레벨 수의 최소화로 설정하였다. Lattice사에서 제공하는 ispLSI1048 칩의 기본 논리 블록은 기본적으로 AND-배열과 OR-배열로 구성되어 있고 AND-배열과 OR-배열을 자유롭게 프로그램해서 원하는 함수를 구현한다. 하나의 기본 논리 블록은 출력 수가 최대 4개, 입력 수가 최대 16개, 적항(product term) 수가 최대 20개까지 구현할 수 있다. 내부에는 플립플롭이 4개가 있는데 이들은 모두 같은 클럭 신호와 리ㅅ 신호를 공유한다. 기술맵핑 과정은 크게 회로 분할과 논리 함수식 추출, 논리 함수의 최소화, 논리 함수식의 그룹핑의 세 개의 과정으로 나뉜다.

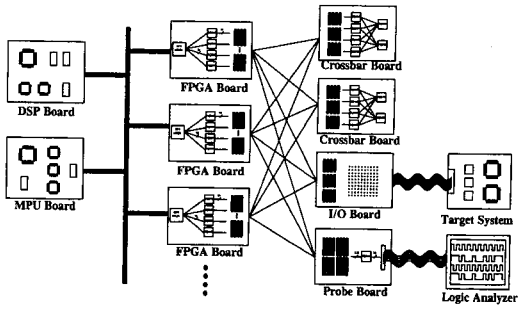
회로의 분할은 ASIC 에뮬레이터에 있어서 성능을 결정 짓는 중요한 과정이다. 분할의 결과에 따라 배치, 배선되어야 할 FPGA 칩의 개수 및 칩간의 연결을 위한 신호선의 수가 결정되기 때문이다. 칩 단위 회로 분할기는 기술 맵퍼에서 출력된 기본 논리 블럭간의 연결 정보, 검증 신호선 및 입출력 신호선 정보등을 고려하여 사용하는 FPGA 칩의 용량에 맞게 분할하며, 특히 에뮬레이션 보드가 갖는 제한 조건을 만족시키면서 FPGA칩간의 연결된 신호선 수의 최소화를 목적 함수로 하였다. 분할 알고리듬은 두 그룹의 분할뿐만 아니라 임의의 여러 그룹으로 분할이 가능한 다중 분할 방법으로 기본적으로 계층 구조적으로 분할이 가능하다.

기본 논리 블록의 연결 정보로부터 에뮬레이션 보드의 구조에 맞게 칩 단위의 분할, 칩 내의 기본 논리 블록의 배치, 칩 내의 핀 할당을 수행한 결과와 각각의 칩에 할당된 기본 논리 블록의 논리 함수식으로부터 칩을 프로그래밍하기 위한 입력 화일인 LDF(Lattice Design File) 화일을 생성한다. 생성된 LDF 화일을 이용하여 각각의 칩에 대하여 Lattice사의 PDS tool을 이용하여 칩내의 배선을 수행한다. 칩 내에서의 완전한 배치, 배선이 끝나면 칩을 프로그래밍할 수 있는 비트 스트림인 JEDEC 화일로 칩을 프로그래밍한다. Lattice사의 FPGA 칩은 시스템내에서 프로그램이 가능하기 때문에 특별한 프로그래밍 장치가 필요 없이 에뮬

레이션 보드에 프로그래밍에 필요한 5개의 핀만 모든 칩에 직렬로 배선한후 모든 칩을 데이지 체인방식으로 연결하여 에뮬레이션 보드상에서 순차적으로 프로그램할 수 있다.

현재 상용화된 에뮬레이터에서 가장 문제가 되는 것은 신호선의 연결에 의한 게이트 이용율에 관한 문제이다. 현재 시판중인 FPGA중 가장 큰 용량을 갖는 FPGA는 10만 게이트 정도이다. 그런데 10만 게이트 용량의 FPGA라 해도 에뮬레이션하는 회로와 FPGA 구조의 특성 차이 때문에 실제 구현할 수 있는 회로의 크기는 최대 용량의 50~60%인 약 5~6만 게이트에 지나지 않는다. 더구나 VLSI의 고집적화로 인하여 에뮬레이션해야 할 회로의 크기는 계속 커지고 있는 상황에서 여러개의 FPGA를 연결할 경우 핀수의 제약 조건 때문에 이용 가능한 게이트수는 20-30%로 떨어지게 된다. 따라서 여러개의 FPGA를 연결할 경우 이용 가능한 게이트수의 감소 뿐만 아니라 칩 사이의 지연 시간이 부가되므로 전체적인 에뮬레이터의 동작 속도는 더욱 느려지게 된다.

그에 반해, 범용적으로 대량 생산되는 마이크로프로세서나 DSP 칩은 FPGA에 비해 상대적으로 훨씬 저렴하고 동작속도도 빠르기 때문에 에뮬레이션하는 회로중에서 DSP나 마이크로프로세서에 의해서 소프트웨어로 처리될 수 있는 부분을 각각의 프로세서에 할당하면 그만큼 에뮬레이션할 게이트수가 줄어들게 되며, 부가적으로 에뮬레이션 속도도 향상되게 된다. 예를 들어 MPEG-2 decoder내의 DCT를 담당하는 블록을 하드웨어로 구성하면 약 8만 게이트정도가 되는데 이것을 에뮬레이터내의 FPGA상에 구현하지 않고 DSP에 의해 전부 소프트웨어적으로 구현하면 적어도 에뮬레이터상에 있는 10만 게이트 용량의 FPGA 2~3개를 절감할 수 있는 효과를 얻는다. 에뮬레이션 속도의 측면에서 볼때, 하드웨어로 처리하던 부분을 소프트웨어로 처리함으로써 야기되는 속도저하를 우려할 수 있겠으나, 원래 에뮬레이터 자체가 real time 에뮬레이션을 지원하기 힘들기 때문에 (최근 상용화된 에뮬레이터의 속도는 2~3MHz정도, 효과적인 에뮬레이션 속도는 약 0.5~1MHz)



〈그림 11〉 개발된 에뮬레이션 시스템(II)

상대적으로 10~20배 정도 빠른 DSP나 마이크로프로세서로 처리하여 수행했을때, 물론 응용 회로에 따라 차이는 있겠지만, 소프트웨어적인 처리로 인한 동작 속도의 문제는 심각하지 않을 것이다.

본 연구실에서 개발한 MPU/DSP가 탑재된 ASIC 에뮬레이션 시스템의 전체적인 하드웨어 구성은 그림 11과 같다.

개발한 ASIC 에뮬레이션 보드는 크게 설계된 회로를 구현하는 logic module, 분할된 회로간의 연결을 담당하는 crossbar module, 그리고 probe module, I/O module로 구성된다. FPGA는 ALTERA사의 EPF10K50칩을 사용하였으며, 배선전용 칩으로는 I-Cube사의 IQ160을 사용하였다. EPF10K50은 5만게이트 정도의 회로를 구현할 수 있으며, IQ160은 모두 160개의 입출력 신호선을 서로 연결할 수 있다. Logic module은 하나의 FPGA와 4개의 배선전용 칩을 사용하였다. 분할된 회로는 FPGA에 구현되고 배선전용 칩은 FPGA의 각 핀에서 나오는 신호선을 각각의 용도, 즉 입출력핀, multiple connection 핀, probe 핀에 따라 구분하여 커넥터에 할당하게 된다. 따라서 FPGA에 회로를 구현할 때 핀 할당 과정을 생략할 수 있어 핀 할당에 따른 FPGA의 이용률 저하를 줄일 수 있다.

분할되어 각 logic module에 구현된 회로 사이의 연결은 crossbar module에서 이루어진다. Crossbar module은 배선 전용칩 6개로 crossbar network을 구성하여 입력 320핀간의 연결을 어떠한 제약없이 수행할 수 있다. Logic module과 crossbar module은 모두 필요에 따라 확장가능하

다. EPF10K50칩 하나의 용량은 5만 게이트로, 실제 약 2만 5천에서 3만 게이트 정도를 구현할 수 있다. 따라서 에뮬레이터에 구현할 회로의 크기에 따라 logic module의 수를 달리할 수 있다. Crossbar module 역시 분할된 회로간의 신호선의 수가 320핀을 하나의 단위로 초과할 때마다 하나의 module을 추가한다. Probe module은 하나의 IQ160 칩과 여러개의 커넥터로 구성되어 128개의 입력중에서 32개의 출력을 선택적으로 뽑아 logic analyzer에 연결하여 신호의 파형을 관찰할 수 있게 설계하였다. 이 probe module 역시 필요에 따라 증가 시킬수 있다. I/O module은 각 logic module로 부터 I/O 신호들을 입력으로 받아 target 시스템과의 인터페이스를 수행한다.

개발한 DSP 보드는 설계된 에뮬레이터와 같이 동작될 수도 있고 stand-alone으로도 동작 가능하다. 데이터의 고속 처리를 위하여 DSP는 33MHz, 32bit로 동작하는 TI의 TMS320C31을 사용하였고 DSP에서 처리된 데이터를 FPGA 보드, MPU 보드, 또는 호스트컴퓨터로 전송할 수 있도록 설계하였다. 또한 메모리의 고속 로딩을 위하여 15nsec로 동작하는 SRAM을 사용하였다. 또한 MPU 보드와의 데이터 입출력은 dual port RAM을 통하여 고속으로 이루어지도록 설계하였다. 프로세서가 동작하기 위한 프로그램은 PC 호스트의 RS-232C를 통하여 DSP 보드의 내부 또는 외부 RAM에 다운로딩되며 처리된 데이터는 address decoder를 거쳐 해당 part로 전송되거나 디버깅을 위하여 다시 host로 uploading될 수 있다. 개발된 보드의 응용 프로그램은 C나 에셈블리어로 작성 가능하며 모든 데이터는 32비트 버스를 통하여 전송된다. 또한 제안된 에뮬레이터가 확장 가능한 multi-board구조이기 때문에, DSP 보드에서 처리된 데이터를 하나의 FPGA 보드가 아닌 복수개의 FPGA 보드 중에서 하나를 선택하여 전송할 수 있도록 하였다.

MPU 보드는 32비트 RISC 프로세서에 의하여 동작하는 시스템으로 메모리와 입출력 기능을 모두 갖고 있으므로 DSP 보드 및 FPGA보드와의 인터페이스는 물론 독자적인 프로그램의 수행 및 확인이 가능하다. 32비트 데이터 버스를 가지며 내부 RAM은 128KB의 충분한 크기를 갖고 있다. 외부와의 인터페이스는 세가지형태로 수행된다. 첫째는 Z8530을 이용한 두 채널 간의 시리얼 데이터 전송에 의한 인터페이스이다. 송수신 데이터는 일반적으로 호스트 컴퓨터와의 인터페이스가 목적이지만 HDLC 프로토콜에 의한 네트웍의 구성이 가능하다. 전송 속도는 동기식일 경우 1MBPS, 비동기식일 경우 11500BPS까지 가능하다. 둘째는 dual-port RAM에 의한 인터페이스로 프로세서간의 인터페이스용으로 개발되었으며 다량의 데이터를 고속으로 송수신할 수 있다. Dual-port RAM은 메모리 셀을 양쪽에서 독립적으로 어드레싱 가능하도록 설계되어 있으므로 multi-processing 회로 구성에도 응용될 수 있다. 셋째로 버스 제어에 의한 인터페이스이다. FPGA 보드 또는 일반적인 하드웨어 로직과의 인터페이스를 하기 위한 것으로 최대 32비트의 데이터 폭을 갖는다. 플립플럽에 의한 ready/busy 논리면으로도 효율적인 데이터 전송이 가능하다.

V. 결 론

회로 설계에 FPGA의 이용이 증가하고 일부 제품에서 FPGA를 적용함에 따라 FPGA에 대한 연구도 활발히 진행되어 다양한 형태의 아키텍춰를 갖는 FPGA가 발표되었다. FPGA는 구현할 수 있는 회로의 크기 및 동작 속도가 제한됨에 따라 응용 범위가 제한되는 단점을 가지는 반면 회로 설계 시간이 단축되고 가격이 저렴하다는 점에 의해 소량 생산될 회로의 제작에 많이 이용되고 있다. 또한 대규모 회로에 대한 검증을 위해 FPGA를 이용한 ASIC 에뮬레이터가 개발되어 회로 검증에 이용되고 있다. 이러한 흐름에 맞추어 본 고에서는 ASIC 설계회로를 빠른 시간내에 구현 및 검증할 수 있는 에뮬레이션 시스템에 대한 개념과 현재 제품화되어 시장에 진출해 있는 상용 에뮬레이션 시스템 그리고 직접 개발한 ASIC 에뮬레이션

시스템을 예를 들어 알아보았다.

국내에서의 에뮬레이터 개발에 대한 연구는 90년대 초반부터 계속되고 있으나 크게 활성화되고 있지 않은 상태이며 아직 실용화되기까지는 좀 더 많은 기간이 요구되고 있다. 연구가 진행되고 있는 곳은 몇몇 대학에 지나지 않으며 그 수준 역시 외국의 상용화된 제품들에 비해 낙후되어 있어 실제 ASIC 설계에 이용되기에는 다소 무리가 따르는 실정으로 외국에서 제품화한 고가의 장비를 수입에 의존하여 사용하고 있다. ASIC 에뮬레이터의 개발에 장기적이고도 지속적인 연구와 투자가 요구된다.

참 고 문 헌

[1] Stephen Walters, "Computer-Aided Prototyping for ASIC-Based Systems",IEEE Design & Test of Computers, pp.4-10, 1991.

[2] Michael Butts, Jone Batcheller and Joseph Varghese, " An Efficient Logic Emulation System", Proc. Int'l Conf. Computer Design, pp.170-177, 1992.

[3] Thomas A. Horvath and Norman H. Kreitzer, " Hardware Emulation of VLSI Design", Proc. 3rd Annual IEEE ASIC Seminar and Exhibit, p13-6.1 - p13-6.4, 1990.

[4] Lattice Data Book, Lattice Semiconductor Corp., 1994.

[5] Virtual ASIC & Concept Silicon, InCA Co, 1993.

[6] Henry Verheyen and Greg Lara, "Rapid Prototyping Using System Emulation Technology", Elecronic Design & Test Conference, 1995.

[7] Quickturn Design Systems, Quickturn Co, 1996.

[8] MPU/DSP Embedded ASIC Emulator, Technical Report, CAD & Communication Circuit Lab, Hanyang University, 1996.

저 자 소 개

鄭 正 和

1950年 3月 10日生

1968年 3月～1975年 2月 한양대학교 전자공학과(공학사)

1975年 3月～1977年 2月 한양대학교 대학원 전자공학과(공학석사)

1978年 4月～1981年 3月 일본 와세다대학교 전자통신공학과(공학박사)

1979年 3月～1980年 12月 일본 NEC 중앙연구소

1986年 9月～1987年 9月 미국 Berkeley 대학 초빙교수

1993年 1月～1994年 12月 대한전자공학회 CAD 및 VLSI 분과 위원장

1993年 3月～1995年 2月 한양대학교 환경과학대학원 GIS 전공주임

1995年 1月～현재 대한전자공학회 교육이사

1995年 8月～현재 한양대학교 전자공학과 학과장

1981年 3月～현재 한양대학교 전자공학과 교수

주관심분야 : VLSI 설계및 CAD, 상위레벨 합성, 논리 합성, 레이아웃 설계, 특히 ASIC 에뮬레이션 시스템 설계, MPEG 회로 설계, 비동기 회로 설계

柳 光 基

1964年 1月 3日生
1982年 3月～1986年 2月 한양대학교 전자공학과(공학사)
1986年 3月～1988年 2月 한양대학교 대학원 전자공학과(공학석사)
1991年 10月～1994年 7月 육군사관학교 전자공학과 교수
～현재 한양대학교 대학원 전자공학과(박사과정 재학중)

주관심분야 : VLSI 설계및 CAD, Layout synthesis, ASIC 에뮬레이션 시스템 설계