특 집

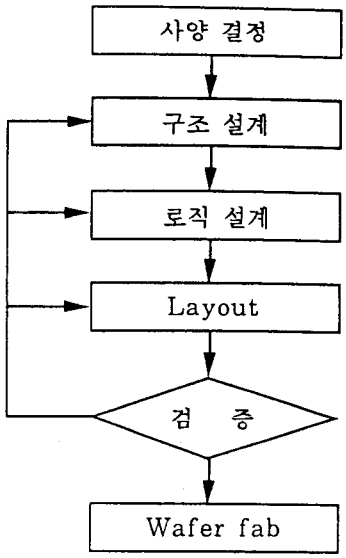
Hardware Emulation 기술

朴 柱 成, 張 豪 根
釜山大學校 電子工學科

I. 서 론

최근 전자산업의 추세는 ASIC(Application Specific Integrated Circuit) 개발이 사업의 성패를 좌우할 정도로 ASIC의 중요성이 날로 강조되고 있다. 이러한 ASIC의 탄생이 가능한 배경을 반도체 제조과정을 나타내는 〈그림 1〉을 통해서 살펴보자. CAD 툴의 발전으로 수작업에 의존하던 구조설계, 로직설계, layout, 검증 등을 획기적으로 짧은 기간에 수행할 수 있어서 개발비용을 상당히 줄일 수 있게 되었다.

지금부터 15년전 국내의 어떤 국가출연연구소에서 4칙 계산용 계산기를 설계하는 프로젝트를 수행한 경험을 소개하면 최근의 VLSI 툴이 설계 엔지니어를 얼마나 획기적으로 지원해 주는가를 알 수 있을 것 같아 잠시 소개하고자 한다. 당시에는 이러한 정도의 LSI를 설계한 경험이 없기 때문에 계산기를 분해해서 칩 내부를 확대한 사진을 통하여 논리회로를 추출하여 전체 회로를 완성했다. 계산기 IC 내부에 있는 ROM의 마이크로 코드 하나를 알기 위하여 2~3일씩을 소비하는 일이 다반사였다. 이러한 어려운 과정을 거쳐 전체 IC의 완전

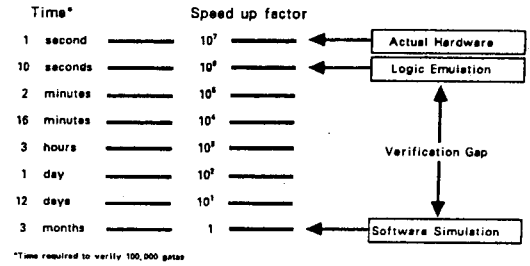


〈그림 1〉 반도체 제조 과정

한 회로를 완성할 수 있었다. 그러나 정작 어려운 일은 설계된 전자회로의 layout 작업이었다. 연구원 2명이 6개월 정도 모눈종이(mylar sheet)에 layout을 하다가 계산기 칩의 개발을 부탁한 회사에 필요한 자료를 구하러 갔더니만 그 회사는 계산기 칩을 구하지 못해서 이미 도산을 하고 말았었다. 만약 당시에 지금과 같은 툴이 지원되었다면 6개월 이상 진행된 layout 작업을 일주일 내에 끝냄으로써 그 회사를 부도에서 구할 수 있었으리라 생각된다.

한편 반도체 제조공정(wafer fab.) 분야에서 장비, 재료, 시설 등의 개선을 통하여 하나의 칩에 집적시킬 수 있는 트랜지스터의 갯수가 매 2년마다 2배씩 증가해 왔다. 또한 전반적인 제조 공정기술의 향상으로 칩의 생산수율(yield)이 높아져서 칩당 가격이 대폭 낮아졌다. CAD툴의 발전으로 설계시간의 단축과 제조 공정기술의 향상을 통한 칩의 생산수율 향상등으로 인하여 비교적 적은 예산으로 필요한 IC를 개발할 수 있게 되어 특정 목적에만 사용되는 주문자 전용의 IC 즉 ASIC의 탄생이 가능했다고 본다.

요즘의 ASIC 개발 추세는 가능한 한 많은 기능을 포함시켜 경쟁사의 시스템보다 우월성을 유지하기 위해 기술이 허락하는 범위내에서 많은 로직 게이트를 하나의 칩에 집적시키고 있다. 80년대 중반쯤에는 1~2만 게이트급의 IC들이 주종을 이루고 있었으나, 지금은 십만 게이트급들의 IC들이 속속 개발되고 있다. 이러한 정도의 IC를 설계함에 있어서 가장 큰 문제는 설계를 검증하는데 시간이 너무나 많이 걸리는 것이다. 예를 들면 〈그림 2〉에서 나타낸 바와 같이 10만 게이트 IC가 실제 시스템에서 1초 동안에 처리하는 모든 일을 워크스테이션에서 완벽하게 시뮬레이션 할 경우 3개월이 걸린다고 주장하는 사람들도 있다.[1] 설사 3개월이 걸리지 않는다고 하더라도 상당히 많은 시간이 소요되는 것은 사실이다. 이와같이 많은 시간이 소요되는 이유는 단순히 개발될 IC 자체의 기능만 체크하는 것이 아니고 주변의 IC들과 시스템 전체 알고리듬을 소프트웨어적으로 모델링하여 수없이 많은 테스트 벡터를 수행시켜야 하고 시뮬레이션



〈그림 2〉 설계검증 방식에 따른 속도 비교

될 때 모든 event들이 순차적으로 진행되기 때문이다.

이러한 문제점들을 보완하기 위하여 하드웨어 모델러를 이용하여 시뮬레이션 시간을 단축하는 경우도 있지만, 여전히 시간적인 문제점을 안고 있다. 반도체 기술이 발전하여 FPGA(Field Programmable Gate Array)라는 새로운 개념의 소자가 개발되어 ASIC의 prototype을 반도체 회사에 의뢰하지 않고 실험실에서 직접 제작할 수 있게 됨에 따라 시뮬레이션 시간을 대폭 줄일 수 있게 되었으며, ASIC의 기능을 사전에 체크해 볼 수 있다. FPGA는 쉽게 설명하자면 백지 로직 IC인데 사용자가 필요한 로직을 썼다 지웠다 할 수 있다. 이러한 FPGA를 이용하여 하드웨어 에뮬레이터가 개발되어 백만 게이트 이상의 집적도를 가진 VLSI도 칩을 제작하기 전에 에뮬레이터를 이용하여 설계를 검증했다.[2] 국내에서도 점차 하드웨어 에뮬레이션의 유용성을 인지해서 ASIC 개발시 설계의 최종 점검을 위하여 에뮬레이터를 이용하는 추세이다.

본고의 II장에서 하드웨어 에뮬레이션의 개념, 에뮬레이션 방식별 특성, 장단점을 짚어 봄으로써 기본 개념을 잡도록 한다. 현재 시판되고 있는 여러가지 제품을 소개함으로써 독자들이 제품의 현황을 파악할 수 있도록 III장을 할애했다. IV장에서는 본 연구실에서 음원 DSP를 하드웨어 에뮬레이션한 예를 보여 줌으로써 에뮬레이션이 실제로 어떻게 진행되는가를 이해할 수 있도록 한다. 마지막으로 V장에서 이 분야에 관련된 몇 가지 생각을 정리한다.

II. 하드웨어 에뮬레이션 특성

1. 에뮬레이션 방식

하드웨어 에뮬레이션 방식을 소개하기 앞서 종래의 ICE(In Circuit Emulation)와 하드웨어 에뮬레이션 차이점을 잠시 언급하고자 한다. 종래의 ICE는 주로 마이크로 프로세서나 DSP를 실제 시스템에 응용하기 위한 소프트웨어를 개발하기 위하여 사용되었다. 지금 이야기 되고 있는 하드웨어 에뮬레이션은 FPGA를 이용하여 개발될 ASIC의 prototype을 제작하여 실제 ASIC의 동작을 확인하는 작업을 의미한다. 시스템의 정상적인 동작을 확인하기 위해서는 하드웨어는 물론 소프트웨어 개발도 해야되기 때문에 하드웨어 에뮬레이션은 종래의 ICE보다 많은 일을 포함하고 있다. 그러나 하드웨어 에뮬레이션은 소프트웨어를 제외한 ASIC의 prototype의 제작으로 해석되는 경우도 있다.

목표 ASIC의 prototype을 제작하는 방식으로 turn-key 하드웨어 에뮬레이터, 프로그래머블 PCB, 사용자 자체제작 보드(custom-built board)를 들 수 있다. Turn-key 하드웨어 에뮬레이터는 prototyping에 필요한 소프트웨어, 디버깅 툴(지원 소프트웨어 및 Logic Analyzer), 몇 십만 게이트를 수용할 수 있는 FPGA 어레이 등의 모든 것을 갖추고 있다. 프로그래머블 PCB는 수만 게이트를 수용하는 FPGA 어레이와 상용의 마이크로 프로세서, DSP, 메모리 등을 하나의 보드에 갖추고 있으며, 이러한 소자들의 연결을 지원하는 소프트웨어가 제공되고 있다. 그리고 제한된 범위이지만 디버깅을 지원하는 소프트웨어와 장치가 있다. 자체제작 보드의 경우는 전적으로 설계자 자신이 FPGA 회사에서 지원하는 툴을 이용하여 개발하고자 하는 ASIC의 prototype을 제작하게 되는데 수만 게이트급 이상의 prototype을 만드는 데는 한계가 있다. 특히 잦은 설계 변경에 따른 래핑 문제와 디버깅의 어려움 등이 문제점으로 대두된다. 그러나 치밀한 공간활용과 설계를 통하여 배선에 따른는 분포용량을 줄일 수 있기 때문에 동일한 FPGA를 사용하게 되더라도 높은 주파수에 동작시킬 수 있다. 이상의 방식들을 여러 측면에서 비교한 것을 〈표 1〉에 정리했다.

〈표 1〉 하드웨어 prototyping 방식별 특성

항 목	에뮬레이터	프로그래머블 PCB	자체 제작 보드
게이트 용량	수십만~3백만 게이트	수만 게이트	?
설계 변경 용이도	좋 음	좋 음	나 쁨
동작속도	-5MHz	5~10MHz	10MHz~
Debugging	아주 좋음	양 호	불 편
가격	억원대	2~3천만원	1~2백만원

2. 장단점

하드웨어 에뮬레이션은 〈그림 2〉에서 나타낸 것과 같이 검증 시간을 획기적으로 단축시키는 것외에도 다음과 같은 여러가지 장점이 있다.

첫째, 반도체 회사에 발주하기 전에 prototype을 실제 시스템에 응용해 봄으로써 설계상의 여러가지 문제점을 발견할 수 있다. ASIC 개발 실패원인 중에서 가장 큰 비중을 차지하는 것이 불충분한 사양으로 인한 오동작으로 전체의 약 50%를 차지한다고 한다. 만약 설계될 IC의 기능이 잘못 정의되어 있다면 아무리 시뮬레이션을 하여도 그 오류는 발견될 수 없을 것이다. 기능이 올바르게 정의되어 있는 경우는 설계자의 잘못으로 인한 오류를 쉽게 찾을 수 있다. 또한 실제 시스템에 응용하여 충분한 테스트를 해봄으로써 시뮬레이션에서 간과하기 쉬운 미세한 문제점을 쉽게 찾아 낼 수 있다.

둘째, 개발될 ASIC에 응용될 소프트웨어 개발 기간을 단축시킬 수 있다. 마이크로 프로세서인 경우를 생각해 보면, 에뮬레이션 시스템이나 prototype이 없는 경우에는 반도체 회사로부터 ASIC이 공급되기 전까지는 본격적인 소프트웨어의 개발은 힘들다. 왜냐하면 시뮬레이션 시간이 많이 걸리고 미묘한 문제를 시뮬레이션을 통하여 찾기는 힘들

기 때문이다. 그러나 에뮬레이션 시스템이 있는 경우에는 반도체 회사에서 ASIC을 제작하고 있는 동안에도 소프트웨어를 개발할 수 있기 때문에 개발기간을 단축할 수 있다.

셋째, 제작된 ASIC에 문제가 발생했을 때 쉽게 문제점을 발견하고 해결할 수 있다. 설계한 IC가 동작하지 않는 경우를 당해본 사람이면 이 문제가 얼마나 심각한 것인지 알 것이다. 원래 IC 내부는 마이크로 미터 폭의 선들이 수없이 얽혀 있기 때문에 어떤 선이 어떤 신호에 해당되는지 알 수 없다. 설사 알 수 있다고 하더라도 신호를 probing 하기가 아주 힘들다. 더우기 요즘의 IC들의 경우에는 적어도 수십개의 신호를 동시에 probing 해야 되기 때문에 동작하지 않는 IC로부터 설계 오류의 힌트를 얻는 것은 불가능하다고 할 수 있다. 하드웨어 에뮬레이션 시스템은 ASIC 내부의 모든 노드 신호를 probing 할 수 있으며 동시에 수백개의 신호를 관찰할 수 있어 문제를 쉽게 파악할 수 있다.

넷째, 기 개발된 IC와 유사한 기능을 가지는 새로운 version을 개발하고자 할 때, 새로운 IC의 설계 검증을 용이하게 할 수 있다. 에뮬레이션 시스템을 통하여 이미 상당부분이 구현되어 있으므로 일부분만 변경하여 실제 시스템에 응용해 봄으로써 추가된 기능의 동작여부와 만족도를 조사해 볼 수 있다. 에뮬레이션 시스템이 없는 경우에는 반도체 회사로부터 새로운 버전의 prototype을 받기 전에는 성능평가를 할 수 없기 때문에 많은 시간과 예산을 낭비하게 된다. 위와 같은 요인들이 모여서 하드웨어 에뮬레이션은 time-to-market을 줄이는데 결정적인 역할을 한다.

단점으로는 동작속도가 늦고, 가격이 비싸고, 사용상의 어려움 등을 들 수가 있다. 속도측면에서 보면 하드웨어 에뮬레이터인 경우에는 범용을 전제로 만들었기 때문에 원래 FPGA보다 훨씬 낮은 속도에서 동작한다. 따라서 실제 시스템과 실시간으로 동작시켜야만 설계를 검증할 수 있는 경우에는 어려움이 따른다. 이러한 경우에는 주변 시스템을 천천히 동작시키거나, 개발될 ASIC의 주요 기능만 체크해 볼 수 있도록 하는 것이 바람직하다. 에뮬레이션 방법이 여러가지 있기 때문에 가격 측면에서는 선택의 폭이 다양하기 때문에 큰 어려움이 없다고 볼 수 있지만, 거대한 칩을 에뮬레이션 하기 위해서는 수십만불 이상의 자금을 투자해야 한다. 그러나 이런 경우는 일단 성공하고 나면 많은 이익이 돌아오기 때문에 투자에 대한 위험을 상쇄 시킬 수 있다. 에뮬레이터의 경우 사용방법이 다소 어렵고 큰 설계인 경우 수십만 게이트의 동작을 일일이 체크해야 되므로 설계검증 단계에서 많은 시간을 소비하게 되어 잘못되면 오히려 turn-around 시간이 늘어날 수도 있다.

III. 상용 에뮬레이터 현황

1980년대 후반에 FPGA가 출현하여 종래의 bread board 방식보다 훨씬 편하게 개발될 ASIC을 검증할 수 있었다. 그러나 개발될 ASIC의 설계가 복잡하여 여러 개의 FPGA로 구현해야 할 때는 수동으로 여러 개의 FPGA로 설계를 나누는 어려움, 각 FPGA 핀들 간 연결 오류 등을 겪게 된다. 그리고 회로를 바꿀 때는 래핑된 것을 풀고 다시 보드를 꾸며야 하는 번거로움이 있다. 이런 문제를 해결하기 위하여 FPGA 어레이를 미리 만들어 두고 프로그램으로 자동으로 나누고 배선하는 방법을 생각하기 시작했다. 어레이에 사용되는 FPGA를 썼다 지웠다 할 수 있는 anti-fuse 형을 쓰면 사용 횟수에 제한을 받지 않아 수없이 많은 ASIC을 에뮬레이션 할 수 있기 때문에 에뮬레이터가 상용화될 수 있었다고 본다. 상용의 에뮬레이터에는 자동으로 나누고 배선하는 주기능에 쉽게 설계를 디버깅할 수 있는 여러가지 기능이 추가되었다. 그리고 디지탈 시스템에서 많이 사용되는 메모리, 프로세서, DSP 등도 어레이에 탑재시켜 설계된 ASIC의 검증을 용이하게 하고 있다.

상용 에뮬레이터는 1991년에 시장에 출하되어 세계 정상의 반도체 회사, 컴퓨터, 통신회사 등에서 널리 사용되고 있다. 현재 국내에 대리점을 두고 있는 에뮬레이터 회사는 Aptix, Quickturm, Zycad사 이다. 본 원고를 시작할 때는 에뮬레이터

의 게이트 용량(capacity), 속도(speed), 기능(function), 가격(cost) 등을 철저하게 분석하여 구매시 도움이 되는 자료를 제공할 계획이었으나, 자칫 잘못하면 오해의 소지가 있을 것 같아 주관적인 평가는 삼가하고 회사에서 제공한 자료를 간략하게 정리하기로 했다.

1. Aptix 제품

이 회사는 Explorer라는 시리즈를 판매하고 있는데 Explorer ASIC, Explorer MP3 Board Emulator가 있다. MP3는 크게 FPCB(Field Programmable Circuit Board), FPIC(Field Programmable Interconnect Component), 지원 소프트웨어로 구성되어 있다.[3] FPCB는 사용자가 미리 만든 FPGA, 상용의 프로세서 , DSP, 메모리 등을 탑재할 수 있는 1,800개의 빈 hole이 있고, 외부와 인터페이스를 위한 각종 커넥터, 필요한 부품이 장착될 소켓들이 있다. 그리고 FPCB 상에는 FPGA들의 핀, 마이크로 프로세서 핀, 메모리핀들을 상호 연결시키는 프로그래머블 스위치 박스 역할을 하는 FPIC가 있어 보드상의 모든 IC를 자동으로 연결한다. MP3는 3×64 신호를 외부의 로직 어날라이저에 공급하여 디버깅이 가능하게 한다. 이 제품은 래핑을 자동으로 해주고 디버깅을 용이하게 probing point를 자유자재로 선택할 수 있게 해주며, FPIC의 연결정보를 PROM 담아서 MP3 보드를 들고 다닐 수도 있다.

Explorer ASIC은 일반적으로 이야기하는 하드웨어 에뮬레이터와 같은 기능을 한다. Xilinx 사의 FPGA, XC4005, XC4010을 각각 5개 , 16개를 사용하여 약 20만 게이트 어레이를 내장하고, 384개의 FPGA 내·외부 신호를 동시에 probing 할 수 있다. 그러나 신호의 파형을 관찰하기 위해서는 타사의 로직 어날라이저를 사용해야 한다. 이 장비에서는 4개의 FPIC를 사용했기 때문에 많은 용량의 FPGA와, RAM, ROM, 마이크로 프로세서, DSP, 자체제작된 ASIC 등을 많이 내장할 수 있다.

2. Quickturn 제품

Qucikturm은 에뮬레이션할 게이트 용량에 따라 Logic Animator, MARS Ⅲ(MP, IP), System Realizer(M250, M3000) 등의 제품을 선보이고 있다. Logic Animator는 50,000게이트 어레이를 내장하고 있으며 큰 설계를 여러 개의 FPGA로 자동으로 나누고, 배선해준다. FPGA 내·외부 신호 448개를 동시에 probing 할 수 있으나, 로직 어날라이저를 내장하고 있지 않기 때문에 타사의 로직 어날라이저를 이용해야 한다. 특히 이 제품은 작은 설계의 에뮬레이션을 목표로 한 제품이기 때문에 connection간의 분포용량을 줄일 수 있어서 대용량 에뮬레이터보다 속도가 빠르다. 다른 특징으로는 FPGA의 configuration을 여러 대의 Logic Animator에 복제시켜 여러 팀이 새로 개발될 ASIC의 성능을 동시에 분석하든지 소프트웨어를 개발할 수 있게 한다.

MARSⅢ는 Quickturn이 93년도에 인수한 PIE의 MARSⅡ를 개선한 것이다. Xilinx의 XC4005를 FPGA 어레이로 사용 했으며 기본적으로 100,000게이트를 지원할 수 있으며 선택에 따라 백만 게이트까지 지원 가능하다. 로직 어날라이저를 내장하고 있어서 다양하게 FPGA 내·외부 신호를 분석할 수 있으며, 1024개의 신호를 동시에 잡아볼 수 있다. ASIC 내부에 있는 작은 용량의 ROM, RAM은 FPGA에 컴파일 시키고 큰 용량은 외부 보드로 지원되게 만들어져 있다. 비디오 신호와 같이 빠른 신호가 에뮬레이션으로 늦게 processing 되는 경우는 처리된 데이타를 모아서 한꺼번에 display 해보게 하는 buffer기능을 갖고 있다.

System Realizer는 M250이 기본 모델인데 Xilinx의 XC4013을 사용해서 250,000게이트를 에뮬레이션 할 수 있으며 M250 모듈들을 추가하여 M3000 모델을 만들 수 있는데, M3000은 3백만 게이트까지 지원할 수 있다. 두 모델 모두 소용량 및 대용량의 메모리 컴파일도 가능하다.[4] 55개 ASIC 라이브러리를 Xilinx 라이브러리로 변환시킬 수 있고, I/O 신호의 갯수는 600개까지 가능하고, 로직 어날라이저 기능을 내장하고 있어서 1152개의 신호를 워크스테이션상에 동시에 잡아볼 수 있다. 물론 이 신호들은 FPGA 내·외부 노드의

신호이다. 그리고 128KB의 depth를 가지는 트레서 버퍼를 가지고 있어 설계상의 문제를 빨리 꺼집어 낼 수 있다.

3. Zycad 제품

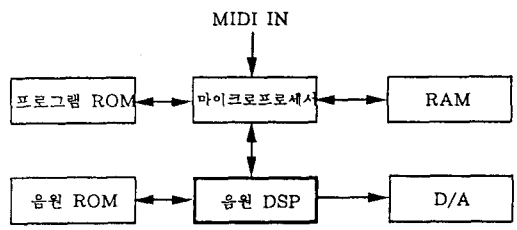
이 회사는 Paradigm RP라는 모델이 있는데 4개의 mother board까지 장착할 수 있다. 하나의 mother board는 Xilinx XC4010 또는 ALTERA FPGA를 이용해서 30,000게이트 어레이를 만들수 있으므로, 총 120,000게이트까지 확장이 가능하다.[6] 하나의 mother board에 8개의 daughter board를 장착시킬 수 있다. 이 보드들에는 대용량 메모리, processor, ASIC, DSP, FPGA 등 사용해서 에뮬레이션에 필요한 보조 시스템을 구성할 수 있다. Xilinx FPGA를 어레이에 사용했을 경우에는 ROM, RAM의 컴파일이 가능하다. 로직 어날라이저는 내장되어 있지 않고 제3 회사의 제품을 이용하여 분석을 해야 한다. FPGA 내·외부 신호를 probing 할 수 있으며, Quickturm 제품과 유사한 데이타 버퍼링 기능을 갖고 있어서 에뮬레이션 속도가 실제 시스템보다 늦을 때 발생되는 문제점을 다소 해결할 수 있다.

IV. 사운드 합성 DSP의 에뮬레이션

부산대학교 VLSI 설계 연구실에서 사운드 합성용 IC 즉 음원 DSP를 개발하는 연구과제를 수행하였다.[6] 이 과제를 수행하는 과정에서 하드웨어 에뮬레이터를 이용하여 목표 DSP의 성능을 확인하고 설계를 디버깅하였다. 이런 과정에서 경험한 몇 가지를 소개함으로써 하드웨어 에뮬레이션에 대한 이해를 돕고져 한다. 프로젝트가 진행됨에 따라 우리가 설계한 DSP가 과연 악기음을 합성할 것인가에 대한 의구심과 불안감이 생기기 시작했다. 그래서 칩을 제작하기 전에 합성음을 들어볼 수 있는 여러가지 방법들을 생각해 보았으나 뾰족한 묘안이 없었다. 그러던 중 하드웨어 에뮬에이터에 대한 정보를 듣고 에뮬레이션을 통하여 우리의 합성음을 미리 듣기로 했다. 또한 합성음을 직접들어 봄으로써 설계상의 문제점들을 해결하려는 것이 우리의 또 다른 목적이었다.

1. 사운드 합성 시스템

대부분의 전자악기 시스템은 〈그림 3〉에서 나타낸 바와 같이 MIDI(Musical Instrument Digital Interface) 신호를 입력으로 받아들여서 원하는 악기음을 합성한다. MIDI 신호는 특정 악기음의 어떤 높이의 음을 얼마 만큼의 세기로 어느 기간동안 연주하라는 정보를 포함하고 있으므로 실제의 합성 데이타에 비해서는 엄청나게 작은 정보이다. 미디 신호는 전자 악기 키보드나 PC의 미디 시퀀서를 통해 음원 모듈로 입력된다. 마이크로 프로세서는 입력되는 미디 신호를 처리해서 악기음 합성에 필요한 파라메터를 계산하고 이를 음원 DSP로 전달하는 역할을 하게 된다. 프로그램 ROM에는 마이크로 프로세서의 동작을 위한 프로그램 뿐만 아니라 음원 ROM에 저장되어 있는 악기음의 샘플 데이타에 대한 각종 정보가 들어 있다. 음원 ROM에는 실제 악기음의 샘플 데이타가 저장되어 있다. 음원 DSP는 악기음 합성 알고리듬을 구현하는 하드웨어 부분으로 마이크로 프로세서에서 제공되는 각종 파라메터를 사용해서 실제 출력될 악기음의 샘플 데이타를 만들어 내는 기능을 수행한다. 음원 DSP에서 나온 악기음의 샘플 데이타는 D/A 컨버터를 통해 아나로그 신호로 변환되어 음원 모듈 외부로 출력된다.



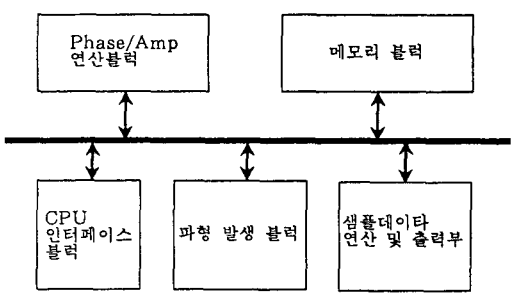
〈그림 3〉 사운드 합성 시스템의 구성도

2. 음원 DSP의 동작

설계된 음원 DSP는 악기음 합성 방식으로 널리 사용되고 있는 Wavetable 방식과 FM 방식에 의

하여 악기음을 합성한다. 그리고 CD급의 음질인 44.1kHz의 샘플링 레이트로 동시에 32개의 악기음을 낼 수 있으며, 좌우 두 개의 채널에 의한 스테레오 효과가 가능하다. 32개의 악기음을 동시에 합성하기 위해서는 32개의 악기음에 해당하는 샘플 데이타 값을 모두 더해서 44.1kHz의 샘플링 레이트로 출력시킴으로써 가능하다. 따라서 하나의 악기음 샘플을 만드는데 할당되는 시간은 0.71μs (1/44.1kHz×32) 정도가 된다. 이것을 하나의 합성 알고리듬을 구성하는 명령어의 갯수로 나누게 되면 동작 주파수가 결정되는데, 설계된 음원 DSP의 동작 주파수는 45MHz이다.

〈그림 4〉에는 설계된 음원 DSP의 내부 기능 블럭도이다. 크게 5개의 기능 블럭으로 나누어져 있음을 볼 수 있다. 먼저 파라메터 메모리와 알고리듬 메모리를 포함하는 메모리 블럭은 마이크로프로세서에서 전달되는 각종 악기음 합성 파라메터와 합성 알고리듬을 구현하는 명령어 코드가 저장되는 곳이다. 파라메터 메모리의 용량은 동시에 합성하려는 악기음의 갯수에 따라 그 크기가 결정되며 알고리듬 메모리는 알고리듬의 갯수, 알고리듬 당 코드의 갯수, 그리고 DSP의 동작 주파수 등에 의해 그 크기가 결정된다. 인터페이스 블럭은 마이크로프로세서와 음원 DSP 간에 이루어지는 데이타 전송을 위한 인터페이스를 담당하는 부분이다. Phase/Amp 연산 블럭은 악기음샘플 데이타가 저장된 메모리의 어드레스를 계산하거나, 악기음의 엔벨로프 변화에 따른 진폭값을 결정하는 부분이다. 파형 발생 블럭은 악기음 합성 방식에 따라 내부 혹은 외부적으로 필요한 파형의 샘플 데이타를 만들어 내는 곳이다. FM 합성을 위해 정현



〈그림 4〉 간략화한 음원 DSP의 내부 블럭도

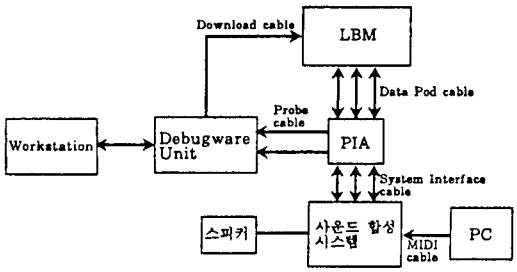
파의 한 주기에 대한 샘플 데이타가 저장된 메모리가 있고, 외부 메모리의 샘플 데이타를 읽어오기 위한 여러가지 제어신호와 어드레스를 발생시키게 된다.

3. 에뮬레이션 시스템

에뮬레이션에 사용된 시스템은 미국의 PIE사의 MARSII인데, 지금은 앞서 소개한 바와 같이 Quickturn이 PIE사를 인수하여 MARSIII를 만들고 있다. 시스템은 LBM(Logic Block Modules), Debugware unit, PIA(Pod Interface Adaptor), Pod 등의 하드웨어와 지원 소프트웨어로 구성되어 있다. 소프트웨어는 설계된 로직의 netlist를 받아들여 하드웨어에 전달하는 과정까지의 모든 작업을 수행함과 더불어 디버깅 시에는 내부 신호에 대한 파형을 워크스테이션 상에 보여준다. LBM은 많은 FPGA로 이루어져 있으며 설계된 로직이 실제의 하드웨어로 구현되는 곳이다. MARSII에서는 Xilinx사의 XC4005를 사용하므로써 RAM이나 ROM과 같은 메모리의 구현이 간단히 이루어질 수 있다. Debugware unit은 MARSII의 하드웨어 기능을 제어하는 역할을 수행하는데 워크스테이션에 탑재된 소프트웨어와 LBM간의 인터페이스를 담당하고, 디버깅 시에는 LBM 내부의 신호에 대한 파형을 디스플레이 할 수 있도록 하는 로직 애널라이저의 기능을 하게 된다. PIA는 설계된 IC가 필요한 시스템 보드 내의 해당 IC 입출력 신호와 IC의 로직이 구현된 LBM을 쉽게 연결할 수 있도록 인터페이스 케이블이 연결되는 곳이다. LBM과 PIA의 연결을 위해서는 전용 케이블이 사용되고, PIA에서 Debugware unit으로 연결되는 케이블을 통해 내부 신호들에 대한 값이 전달된다. Pod는 PIA와 LBM간, PIA와 Debugware unit을 연결하는 긴 케이블의 양 끝에 붙어서 강력한 신호를 driving 해주거나 약한 신호를 shaping 해 준다.

4. 음원 DSP의 에뮬레이션 과정

〈그림 5〉는 설계된 음원 DSP의 에뮬레이션을 위한 시스템 구성도를 나타내었고, 〈그림 6〉은 이



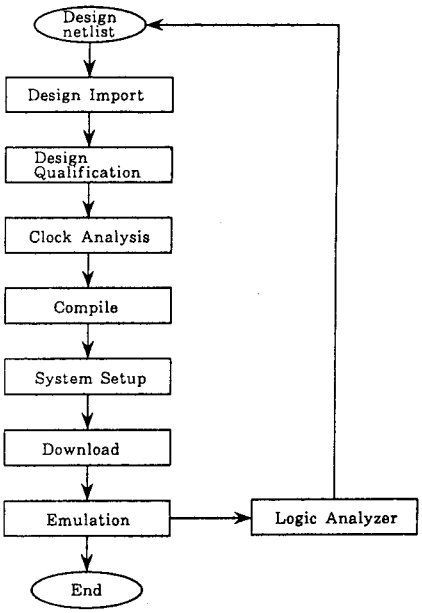
〈그림 5〉 음원 DSP의 에뮬레이션을 위한 시스템 구성도

〈그림 6〉 음원 DSP 에뮬레이션 장면

에 따라 구성된 실제 실험 장면을 보여주고 있다. 그림과 같은 시스템 구성하에 PC의 미디 시퀀서를 통해 음악을 연주하면 LBM에서 구현된 음원 DSP를 통해 악기음들의 샘플 데이타가 만들어지고, 보드 상의 D/A 컨버터를 통해 아날로그 파형으로 변환된 소리를 스피커를 통해 직접 들어봄으로써 여러가지 악기음에 대해 완벽한 합성이 이루어지는지를 살펴보게 된다.

원래 설계된 음원 DSP는 32개의 악기음을 동시에 합성하고 이를 위해 필요한 동작 주파수는 45MHz였다. 그러나 실제로 로직 에뮬레이터가 동작하는 속도는 수 MHz 이내로 되어 한정된다. 따라서 우리의 경우는 음원 DSP의 에뮬레이션을 위해서 안전하게 3MHz 정도의 클럭 주파수를 사용하기로 하였고, 이로 인하여 동시에 합성할 수 있는 악기음의 수가 2개로 제한되어 버리지만 전체 설계 개념을 검증하기에는 충분하다.

〈그림 7〉은 MARSII를 이용한 음원 DSP의 에



〈그림 7〉 음원 DSP의 에뮬레이션 순서도

뮬레이션에 대한 전체 순서도를 보여준다. 이 순서에 따라 단계별로 어떤 작업들이 이루어지는가를 살펴보면 다음과 같다. 처음에 음원 DSP의 설계는 ASIC 설계 툴인 COMPASS 상에서 게이트 레벨의 디자인에 의해 이루어졌다. 설계된 데이타를 MARSII에 입력시키기 위해서는 MARSII의 netlist 입력 형태의 하나인 EDIF 형태로 변환시키는 작업이 ASIC 설계가 이루어진 COMPASS에서 행해진다. EDIF 형태로 변환된 netlist는 Design Import 단계를 통해 워크스테이션 내에서 해당 데이타 베이스를 형성하게 된다. Design Qualification 단계에서는 입력된 netlist에 대한 검증이 이루어져 설계 오류와 게이트 수에 대한 정보가 나오고 에뮬레이션 시스템에서 수용 가능한 것인지의 여부가 결정되어진다. 다음 단계는 Clock Analysis로서 MARSII에서는 특별히 지연시간이 문제가 되는 신호들을 위해서 super buffer를 사용하므로 이것을 효율적으로 사용하기 위해 클럭 정보에 대한 명시가 필요하다. 또한 이 단계에서 critical net에 대한 지정, 셀 그룹핑과 같은 작업이 이루어진다. 그 다음 단계로서 Design Compile이 이루어지는데 이 과정을 통해 LBM 내에 있는 FPGA들

에 대한 configuration 정보와 FPGA들간의 연결 정보가 나오게 된다. 이 정보들은 Debugware Unit을 통해 LBM으로 직접 다운로드 된다. 해당 로직에 대한 컴파일이 끝나고 나면 System Setup 단계에서 PIA와 LBM간의 연결, PIA와 Debugware unit과의 연결, PIA와 실제 시스템과의 연결이 이루어진다. 모든 연결이 끝나고 나면 워크스테이션으로부터 Debugware를 통해 LBM으로 다운로드가 행해지고, 그 때부터 음원 DSP의 실제 동작을 살펴보고 로직 애널라이저 기능을 통한 디버깅 작업 등이 이루어지게 된다.

5. 에뮬레이션을 통한 디버깅

실제 에뮬레이션을 통해 많은 디버깅 작업이 이루어졌는데 결정적인 두 가지를 소개하면 다음과 같다. 처음에 악기음을 합성시켜 보았을 때는 합성된 음이 부분적으로 깨어지는 현상을 발견하였다. 음의 처음 부분은 소리가 잘 나다가도 갑자기 삑하는 소리가 난다거나 음이 끊어지는 것이었다. 음원 DSP가 음을 합성하는데 가장 먼저 하는 일은 마이크로 프로세서로부터 합성에 필요한 파라메터를 전달받는 것이므로 먼저 이 동작이 올바르게 이루어지고 있는지를 살펴보았다. 에뮬레이터를 이용하여 DSP 내부의 레지스터들에 있는 내용을 조사해 본 결과 마이크로 프로세서와 DSP 간의 인터페이스에 문제가 있다는 것을 발견하게 되었는데, 그것은 음원 DSP의 에뮬레이션을 위해 DSP의 동작 주파수를 45MHz에서 3MHz로 다운시킴에 따라 발생된 문제였다. 마이크로 프로세서는 실시간으로 미디 신호를 처리하고 있는데 DSP의 동작 주파수를 낮추어 버림으로써 DSP가 마이크로 프로세서에서 주는 데이타를 제대로 받아들일 수 없었기 때문이었다. 마이크로 프로세서의 동작을 적절하게 조절하므로써 이 문제를 해결할 수 있었다. 개발될 IC가 전체 시스템에 종속적으로 동작하면서 실시간 신호처리를 해야 할 경우에는 이와 유사한 문제점들이 발생할 수 있다.

그 다음 문제는 악기음이 나는 도중에 계속 듣기 싫은 찌찌직거리는 잡음이 합성음에 실려서 들리는 것이었다. 이 잡음의 원인을 추적하는 데는 많은 시간을 소비하였다. 왜냐하면 음원 DSP의 명령어가 제대로 수행되고 있는지의 여부에서부터 에뮬레이션 시스템을 연결하는 케이블에 발생하는 노이즈의 영향까지를 모두 따져봐야 했기 때문이다. 내부 로직을 살펴보는 데는 에뮬레이터에서 제공되는 로직 애널라이저의 기능이 많은 도움이 되었다. 이를 통해 명령어의 올바른 수행 여부를 내부 신호에 대한 파형을 살펴보므로써 확인할 수 있었다. 많은 시행착오 끝에 음원 DSP 내부의 accumulator에서 외부 D/A 컨버터로 전달되는 샘플 데이타의 값이 어느 순간 갑자기 커져 버리는 것을 발견하였고, 그 원인은 곱셈기를 포함한 샘플 데이타 연산부가 특수한 경우에는 잘못 동작하는 것이었다. 이 문제를 고침으로써 마침내 음원 모듈이 지원하는 모든 악기음에 대해 완벽한 합성이 이루지는 것을 확인할 수 있었다.

V. 결 론

본고를 통하여 하드웨어 에뮬레이션의 장점과 단점, 현황, 그리고 실제 에뮬레이션 예를 살펴보면서 하드웨어 에뮬레이션이 time-to-market을 줄이는 방법이 된다는 것을 알 수 있었으리라 본다. 마지막으로 이 분야에서 생각해야 할 몇 가지를 언급하고자 한다.

첫째, 에뮬레이션 하고자 하는 시스템을 정확하게 분석해야 한다. 물론 ASIC을 설계하기 위해서 ASIC의 동작 환경을 잘 파악했겠지만, 에뮬레이션을 위해서 전체 시스템을 수정할 필요가 있기 때문에 시스템 동작을 보다 심도있게 파악해야 할 필요가 있다. 이러한 문제는 대부분 에뮬레이터의 동작속도가 실제 하드웨어보다 느리기 때문에 발생된다.

둘째, 가장 간단한 동작부터 에뮬레이션 하는 것을 권하고 싶다. 에뮬레이터 제품 소개서를 보면 모든 종류의 로직 구현이 가능한 것으로 되어 있고, 게이트 용량도 풍부하고, 사용방법도 별로 어렵지 않게 보이기 때문에 단번에 전체 기능을 에뮬

레이션 하고 싶어질 것이다. 그러나 에뮬레이션 자체가 그렇게 쉬운 일이 아니기 때문에, 일이 잘 안 풀릴 때는 설계 잘못인지 에뮬레이션 방법이 틀린 것인지 모르는 수가 있다. 일단 중요하지 않은 기능 블럭을 과감하게 버림으로써 설계 분석이 용이하게 하여 문제에 접근하는 것이 좋다. 일단 성공하고 나면 나머지 기능을 추가하는 것은 어렵지 않다.

셋째, 에뮬레이션 전문가를 양성시키는 것을 권하고 싶다. 장비 사용법이 그렇게 간단하지 않고 에뮬레이션 작업이 잡다한 일들을 많이 요구하기 때문에 초보자일 경우에는 상당한 시간이 걸린다. 전담자가 있을 경우는 장비의 분석 기능을 효율적으로 사용할 수 있고 작업과정 전체 흐름을 알고 있기 때문에 설계자와 에뮬레이션 전문가가 공동으로 수행하면 아주 짧은 기간에 에뮬레이션을 성공할 수 있을 것이다.

넷째, ASIC 개발과 관련이 있는 공공기관에서 에뮬레이션 서비스를 해주는 것이 좋겠다. 앞으로는 중소기업의 ASIC 개발이 상당히 많아질 것이다. 중소기업의 경우에는 한번 ASIC 개발에 실패하면 회사가 망하는 경지까지 이를 수가 있다. 그리고 에뮬레이터 장비가 비싸고 그렇게 자주 쓸 장비가 아니기 때문에 중소기업에서 구입하기가 어렵다. 이러한 문제점들을 해결하기 위하여 국가출연연구소나 학교에서 적정한 가격으로 서비스 해주는 것을 생각할 수 있다. 이러한 방법이 어렵다면 장비 대리점에서 이러한 역할을 대신 하는 것도 고려할 수 있을 것이다.

다섯째, 에뮬레이터의 기능이 보다 개선되어야 한다. 문제가 되는 것들로는 속도, 라이브러리 호환문제, 툴간의 인터페이스 등을 들 수 있는데 그 중에서 가장 문제가 되는 것은 속도이다. 에뮬레이션할 ASIC이 시스템으로부터 오는 신호를 실시간으로 처리해서 되돌려 주어야 할 경우는 에뮬레이션이 불가능한 경우도 있다. 이러한 경우가 아니고 디스플레이 되기 전의 비디오 신호와 같은 일방통행식의 신호인 경우에는 문제가 덜 심각하다. 그러나 이런 경우에도 실제 상황의 수 초에 해당되는 테스터 벡터를 만드는 것이 불가능하므로 입력 테스터 벡터를 실제 시스템으로부터 받아 두었다가 에뮬레이터가 감당할 수 있는 속도로 신호를 처리하는 기능이 있으면 아주 편리할 것이다.

여섯째, 국내에서도 에뮬레이터를 개발해서 싼 가격으로 공급했으면 한다. 지금 우리나라에서 사용되는 에뮬레이터는 모두 미국산인데 너무 비싸기 때문에 대학이나 중소기업 같은 데는 엄두도 낼 수 없다. 처음부터 외국 것과 같은 다양한 기능을 생각하지 말고 PCB에 FPGA 어레이를 만들어 두고 자동으로 partition과 routing되는 정도의 제품을 개발해도 어느 정도 시장은 있으리라고 본다.

감사의 글

본고의 음원 DSP 에뮬레이션을 위하여 에뮬레이터를 사용하게 해준 서두로직(주)와 전자부품종합기술연구소에 감사를 표한다. 그리고 음원 DSP 에뮬레이션은 과학기술처와 한국과학재단의 연구비 지원(과제번호 : 94-0100-12-01-03)에 의하여 수행되었다.

참 고 문 헌

[1] Leenong Li, "Application for Reprogrammable Emulation Technology," Electronic Design Automation & Test Conference, 1995.

[2] James Gately, et al, "UltraSPARCTM-I Emulation," Design Automation Conference, 1995.

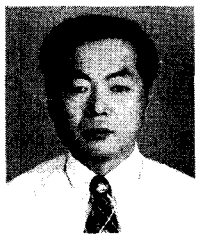
[3] Henry Verheyen and Greg Lara, "Rapid Prototyping Using System Emulation Technology," Electronic Design Automation & Test Conference, 1995.

[4] Ruey-sing Wei, et al, "Synthesis for Emulation," Electronic Desing & Test Conference, 1994.

[5] Dave Aiken, "Rapid Prototyping Systems," Electronic Design Automation & Test Con-

fer-ence, 1994.
〔6〕 박주성 외 다수, "전자악기용 디지탈 신호처리 VLSI 개발," 연구보고서, 과학기술처, 1994

저 자 소 개

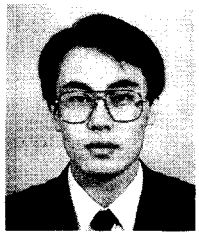


朴 柱 成

1953年 12月 19日生
1976年 2月 부산대학교 전자공학과 졸업(학사)
1978年 2月 한국과학기술원 전기 및 전자공학과(석사)
1989年 8月 University of Florida 전자공학과(박사)

1978年 3月～1985年 8月 한국전자통신연구소 연구원, 선임연구원, 반도체 설계개발실장 역임
1985年 8月～1989年 8月 University of Florida 연구조교
1989年 9月～1991年 2月 한국전자통신연구소 집적회로 연구부 연구위원
1991年 3月～현재 부산대학교 공과대학 전자공학과 교수

주관심분야 : DSP 설계 및 응용, Sound 합성, 반도체 소자 모델링



張 豪 根

1968年 1月 15日生
1993年 2月 부산대학교 전자공학과 졸업(학사)
1995年 2月 부산대학교 대학원 전자공학과(석사)
1995年～현재 부산대학교 대학원 전자공학과 박사과정

주관심분야 : DSP 설계 및 응용, Sound 합성