特輯

32bit RISC 그림돌 시스템의 동시 공학적 설계

慶 宗 旻
韓國科學技術院 電氣 및 電子工學科

Ⅰ. 서 론

최근의 VLSI 제조 기술의 급속한 발달은 과거에 여러개의 칩으로 보드상에 구현되는 것을 한개의 칩 속에 구현될 수 있게 해 주고 있다. 그래픽이나 전자 출판 시스템같이 특정 용도의 응용 분야에서는 이러한 집적도가 코아부분 뿐만 아니라 주변 디바이스등을 단일칩화 해 줄 수 있기 때문에 저가격대의 고성능 시스템 구현에 대한 효과적인 설계를 제공해 줄 수 있다. 이러한 것을 동기로 하여서 과학기술원에서는 RISC형 마이크로프로세서 코아를 근간으로 하고, 관련 주변 디바이스(host interface controller, DRAM controller, video system controller등)를 단일칩화 하여서 그래픽 시스템 혹은 전자 출판 시스템등의 응용 영역을 지향하는 저가격/고성능의 그림돌 및 관련 시스템의 개발에 박차를 가해 왔다.

그러나 마이크로프로세서 및 관련 시스템의 설계는 여러 분야에서 다양한 사람들의 노력을 요구한다. 예를들면 그것이 칩 자체의 설계 뿐만 아니라 컴파일러 그리고 응용 보드 및 응용 소프트웨어 등이 제대로 결합되어야만 최종적으로 완전한 성능을 낼 수 있기 때문이다. 이때 중요한 사실은 각 분야에서 해야 할 일들이 서로 독립적인 것들이 아니고 서로가 최종 기능 및 성능에 결정적으로 영향을 줄 만큼 유기적으로 연관이 되어있다는 사실이다. 그래서 마이크로프로세서 및 관련 시스템의 설계와 같이 방대한 일을 어떻게 효과적으로 분할한 후에 이를 최종적으로 결집할 것인가 하는 것이 대단히 중요한 사항이 된다. 그러므로 이러한 사항들이 대규모 시스템의 동시 공학적인 설계 측면에서 수행되는 것이 매우 바람직하다.

본 논문에서는 32 bit RISC microprocessor인 그림돌 칩 및 관련 시스템 개발이라는 예제를 통하여 상기의 동시 공학적인 설계 방식을 설명하고자 한다. 이는 응용영역을 고려한 칩의 구조 및 명령어 설계를 출발점으로 하여서 크게 칩 자체 설계, 어셈블러 및 컴파일러등의 시스템 소프트웨어 설계, 응용 프로그램 시뮬레이터를 통한 응용 소프트웨어 설계 그리고 응용 보드 시뮬레이터를 통한 응용 보드 설계등으로 분할되어서 동시에 설계가 된 후에 최종적으로 전체의 응용 시스템으로 결집되어가는 과정을 포함한다. 이를 위해서 먼저 Ⅱ 장에서는 그림돌 및 관련 시스템의 최종 설계 목표를 위해서 동시 공학적인 측면에서 전체적인 설계 방법론에 대해서 설명한다. Ⅲ 장에서는 칩, 소프트웨어 그리고 응용 보드 각각에 대해서 설계 방법론 및 설계된 내용 그리고 서로의 유기적인 관계에 대해서 언급한다. Ⅳ 장에서는 최종적으로 완성된 응용 시스템에 대한 측정 결과 그리고 칩 내의 결함 발견 및 제거 방법등에 대해서 언급한다. 마지막으로 Ⅴ 장에서는 본 시스템 설계의 경험에 의하여 앞으로 개선의 여지가 있는 사항들에 대해서 언급한다.

Ⅱ. 동시 공학적인 측면에서의 설계 방법론

그림 1은 전체적인 설계 방법론을 나타내고 있는데 크게 specification 설계단계, behavior 모델링 단계, 그리고 physical 설계 단계등이 그것이다. specification 설계 단계에서는, 주어진 응용 영역을 충분히 만족시킬 수 있는 칩의 구조 및 명령어 설계가 주로 행하여 졌다. 특히 명령어 설계는 일차적으

로 서로 독립적인 설계 활동, 즉 칩에 대해서 VHDL를 이용한 하드웨어적인 모델링과 ISS(명령어 시뮬레이터 : Instruction Set Simulator)의 출발점이기 때문이다. 이러한 명령어 및 구조 설계를 위해서 기존의 응용 소프트웨어등(PostScript Interpreter 등)의 효율적인 수행을 위해서 필요한 명령어 조사, 핵심적인 그래픽 루틴, 그리고 가격과 성능면에서 필요한 하드웨어등이 무엇인가에 대해서 연구하였다. 이렇게 하여 정해진 명령어 및 하드웨어 구조는 다음 단계의 behavior 모델링 단계로 넘어가게 하였다.

behavior 모델링 단계에서는 서로 독립적이면서 유기적 관계가 있는 두가지 설계 활동이 진행되었다. 첫째는 C 언어로 구현된 ISS로서 주로 그림돌의 성능 측면에서 명령어 세트와 구조 검증에 사용되었다. ISS의 수행결과는 어떤 프로그램 루틴이 수행되는데 사용된 명령어에 대한 통계적 자료, 전체 수행 사이클등 그림돌의 성능을 짐작할 수 있게 하기 때문에 이를 근거로 명령어 세트와 구조에 대한 수정이라는 과정들이 가능하였다. 둘째는 VHDL를 이용한 칩의 하드웨어 모델링으로서 이는 주어진 하드웨어 구조 내에서 설계된 명령어 세트가 문제없이 수행되는지를 검증하는데 이용되었다. 이때 중요한 사실은 ISS와 VHDL 모델이 똑같은 동작을 해야 한다는 것이다. 이 양자간의 동일성은 어셈블러를 통해서 이루어졌다. 즉 모든 명령어에 대해서 그리고 가능한 그들의 모든 조합으로 이루어진 어셈블러 프로그램이 양쪽에서 똑같은 기능을 하는지를 검증함으로써 양자간의 동일성을 보장할 수가 있다. 이 과정은 세번째 단계에서 일어날 또 다른 독립적인 설계활동을 위해서 꼭 필요한 일이었다. 이후에 소프트웨어 그룹에서는 어셈블러에 이어 링커와 로더와 아울러 컴파일러 시스템도 개발하였다.

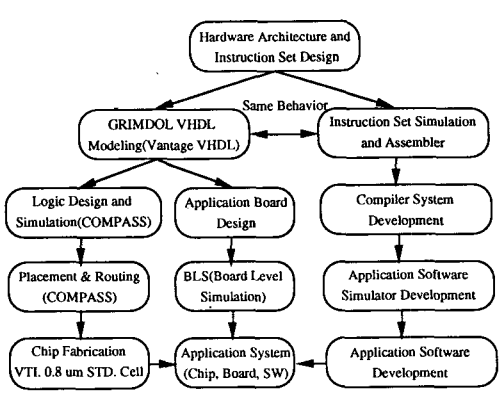

그림 1. 동시 공학 측면에서의 설계 방법론

physical 설계 단계에서는 세가지의 서로 동시적인 설계 활동이 동시에 일어났다. 이는 behavior 모델링 단계에서 완성된 시스템 소프트웨어(ISS, 어셈블러, 컴파일러)와 VHDL 모델를 근거로 하고 있다. 첫째는 그림돌의 로직 설계로서 이는 VHDL 모델를 근거로 하여 COMPASS 회사의 ASIC(Application Specific Integrate Circuit)전용 설계툴인 Design Navigators 하에서 0.8 마이크론 표준셀 라이브러리를 가지고 설계하였다.

둘째는 응용 보드 설계로서 Mentor 회사의 Design Architect 툴 하에서 행하여졌다. 이를 통하여 응용 보드 설계에 필요한 TTL 라이브러리와 그림돌 VHDL 모델이 함께 시뮬레이션될 수 있었다는 것이다. 이것은 실제 응용 보드에서의 타이밍까지도 고려한 시뮬레이션이기 때문에 그림돌 칩이 나오기전에 응용 보드를 설계할 수 있는 장점이 있다.

셋째는 응용 소프트웨어 설계이다. 이는 두번째 단계에서 설계된 시스템 소프트웨어인 ISS와 어셈블러를 포함하는 컴파일러에 기초를 두고 있다. 즉, 기설계된 ISS를 실제 그림돌 프로세서가 장착되어 동작하는 보드까지 시뮬레이션할 수 있도록 확장된 응용 프로그램 시뮬레이터를 통하여 작성된 응용 소프트웨어가 제대로 작성되었는지를 검증하는 것이다. 이 응용 프로그램 시뮬레이터는 어셈블러로 작성된 프로그램이나 C 언어로 작성된 응용 프로그램이 컴파일된 결과인 binary 화일을 입력으로 받아서 ISS상에서 동작을 한 후 그 결과를 X-Window상에서 display할 수 있도록 된 것이다. 이러한 응용 프로그램 시뮬레이터를 통하여 사전에 응용 프로그램의 상당량을 설계해둠으로써 추후에 응용 프로그램 포팅을 단시간 내에 가능하게 하였다.

physical 단계에서의 3가지 독립적인 설계 활동의 결과 그림돌 칩, 응용 소프트웨어 그리고 응용 보드는 칩이 나온 2일 이내에 전체 시스템으로서 완성이 되었고 이 시스템이 전체적으로 성공적으로 동작하는 것을 검증하기까지는 일주일이 걸리지 않았다.

Ⅲ. 그림돌 및 관련 시스템의 설계

본장에서의 Ⅱ 장에서 설명된 설계 방법론의 마지

막 단계인 설계 단계에서의 3가지 동시적인 설계 활동 즉, 그림돌 칩 설계, 응용 보드 설계 그리고 응용 소프트웨어 설계를 중심으로 상세한 내용을 설명하고자 한다.

1. 그림돌 칩 설계

그림돌 칩은 X-Y 좌표계 연산, 그리고 8 bit 단위의 픽셀 데이타 처리등 2차원 그래픽 동작들과, Post Script Interpreter의 효율적 수행에 적합하도록 설계된 형태의 마이크로프로세서이다. 이외에도 그림돌 칩은 실제 응용 시스템에서 요구되는 DRAM controller, host interface controller 그리고 video system controller등의 주변 디바이스들을 단일칩으로 구현함으로써 응용 시스템의 가격 및 성능 향상을 추구하였다. 그림2는 그림돌 칩의 전체 블락 다이아그램이고 그림 3은 제작된 그림돌 칩 사진이다.

그림돌의 RISC 코아부분은 5단계의 파이프라인 구조를 가지고 있는 load/store machine으로서 모든 메모리 억세스는 load/store 명령어를 통해서 이루어진다. 마이크로프로세서의 특징인 고정된 명령어 길이와 고정된 op-code의 위치는 명령어 해석기를 간단히 함으로써 하드웨어가 단순해진다는 장점이 있다. [1] [2] 내장된 2 Kbyte(512개의 명령어) 캐쉬는 명령어에 의한 메모리 억세스와 명령어 페치간의 어드레스/데이타 버스의 resource conflict를 없앰으로써 파이프라인이 stall되는 것을 막을 수 있다. 파이프라인의 성질에 의해서 생기는 data conflict도 internal forwarding [2] 에 의해서 해결하였다.

그림돌 칩의 데이타 패스 부분은 크게 5개의 부분으로 구성되어 있다. 즉 ALU, 16 x 16 Booth Multiplier, Funnel Shifter 그리고 32개의 Register File이 그것이다. ALU 내에 있는 32 bit adder는 16 bit 단위의 데이타 처리를 가능하게 함으로써 16 bit 단위의 X 좌표 그리고 Y 좌표간의 좌표 계산을 효율적으로 할 수 있다. 16 x 16 Booth Multiplier는 고품위의 외곽선 문자를 구성하는 Bezier 곡선 [3] [4] 처리에 꼭 필요한 기능 블락으로서 이러한 multiplication을 단순히 adder와 shifter 기능으로만 실행하면 많은 사이클의 소요로 성능이 저하된다. Funnel Shifter와 Merger 유닛은 32 bit 데이타중에 임의의 필드 데이타를 추출 혹은 삽입할 수 있는 기능 블락으로서 이는 픽셀 단위의 데이타 처리를 위해서 꼭 필요하다.

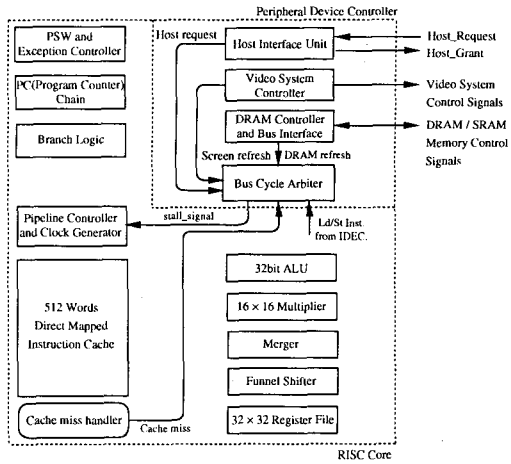

그림 2. 그림돌 칩 블락 다이아그램

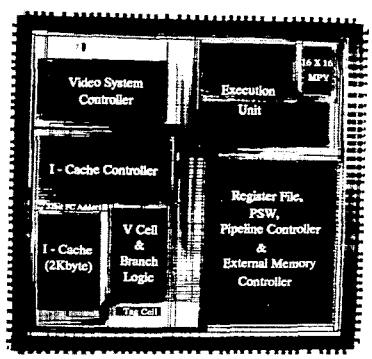

그림 3. 그림돌 칩 사진

그림돌 칩에서 또 다른 중요한 블락은 DRAM controller, host interface controller 그리고 video system controller이다. 그림돌 칩이 지향하는 응용 시스템은 대부분 외부 메모리로서는 DRAM을 쓰고 아울러 host system으로부터 코드, 데이타 등을 down-load 받거나 동작중에 command level의 정보를 주고 받아야 한다. [5] [6] 아울러 스크린등의 video system과 결합되므로 VRAM(dual port DRAM)에게 screen refresh 시간을 정확히 알려주는 video system controller등이 칩 내에 있으면 추구하는 전체 시스템의 가격 및 성능향상에 큰 도움이 된다.

그림 4는 그림돌 칩의 전체적인 설계 방법론에 대한 것이다. 먼저 specification 설계 단계에서 정의된 하드웨어 구조 및 명령어 세트를 근거로 이를 하

드웨어 기술 언어인 VHDL로 그림돌 칩을 모델링하였다. 이때 효율적인 VHDL 모델링을 위해서 칩내의 하드웨어들을 소단위의 기능 블락들로 먼저 나눈 후 이 기능 블락에 필요한 버스 및 control 신호들을 정확히 정의하였다. 이때 중요한 사실은 소단위의 모델링에 대해서 충분한 시뮬레이션을 거쳐야 된다는 것이다. 왜냐하면 이 소단위에서 설계오류가 없어야 이 소단위를 묶은 전체 시뮬레이션에 설계 오류가 줄어들기 때문이다.

이렇게 하여 기술된 그림돌 칩에 대한 VHDL 모델은 VTI사의 0.8 마이크론 표준셀 라이브러리를 이용한 로직 설계의 근간이 된다. 왜냐하면 소단위별로 정의된 기능, control 신호 그리고 버스 신호등이 VHDL 시뮬레이션에 의해서 모두 검증이 되었기 때문이다. 로직 설계를 한 후 역시 소단위 별로 충분한 로직 시뮬레이션을 하였다. 이때 로직 설계에 사용된 설계 툴은 schematic entry, 데이타 패스 컴파일러, 셀 컴파일러(메모리, multiplier 등), 로직 합성기 그리고 ASIC 합성기[7] 등 설계 블락에 따라 여러가지 적합한 툴을 사용하였다. 이렇게 하여 설계된 전체 로직에 대해서 다음과 같은 시뮬레이션 환경을 이용하였다. 먼저 손으로 작성된 어셈블러 프로그램이나 컴파일러가 생성한 binary 화일을 가지고 VHDL 시뮬레이션을 한다. 그리고 나서 그림돌 칩의 외부핀에 발생된 신호를 매 clock 사이클마다 두번 샘플링하여서 이를 table 화일 형태로 만든 후 이를 다시 COMPASS의 로직 시뮬레이터인 QSIM이 받아 들일 수 있는 입력 stimulus로 변환시킨 후 이를 가지고 전체 로직 시뮬레이션을 행하였다. 이러한 방식의

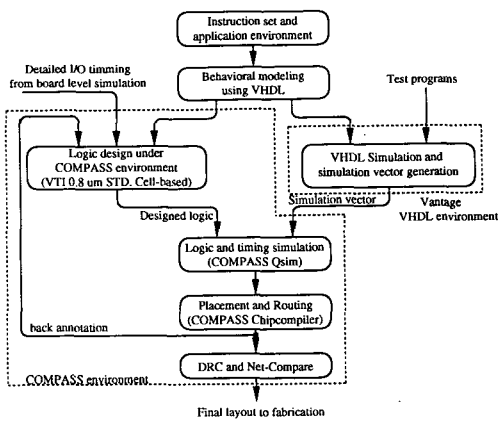

그림 4. 그림돌 설계 방법론

장점은 사용자 환경이 좋은 상위 시뮬레이터에서 많은 시뮬레이션을 한 뒤 이 결과를 로직 시뮬레이션에 똑같이 적용함으로써 충분한 검증를 거친 VHDL 모델과 동일한 기능을 하는 로직 설계를 얻을 수 있다는 것이다. 이후 로직 시뮬레이션으로 검증된 전체 로직들은 표준셀 방식의 배치 및 배선을 한 후 parasitic을 추출한다. 추출된 parasitic을 다시 netlist에 포함 시킨 후 back annotation 시뮬레이션을 함으로써 최종적인 검증을 거친 후 DRC(Design Rule Check)와 netlist compare를 거친 후 그림돌 칩 제작에 넘겨졌다.

2. 시스템 및 응용 소프트웨어 설계

그림돌 칩 및 관련 시스템 설계에서 언급될 수 있는 중요한 부분은 소프트웨어 부분이다. 이는 주로 ISS, 어셈블러, 컴파일러 그리고 응용 소프트웨어이다. 여기서 또 다시 언급이 되어야 할 부분은 ISS이다. ISS의 원래 목적은 설계된 명령어들의 올바른 수행여부 검증과 이 명령어들로 이루어진 여러 프로그램들을 실행시킨 후에 사용된 명령어 빈도등의 통계적 데이타나 수행 사이클들을 조사해 봄으로써 명령어 및 하드웨어 구조 조정에 대한 iteration loop를 제공하기 위함이었다. 이러한 ISS가 그림돌 칩과 똑같은 behavior를 갖는다는 점을 이용하여 응용프로그램을 사전에 시뮬레이션한 후 최종적으로 응용 보드상에서 올바른 결과를 낼 수 있는지를 알수 있는 응용 프로그램 시뮬레이터로 확대 설계되었고 이는 응용 프로그램 개발에 커다란 공헌을 하였다.

ISS는 DLX 시뮬레이터[8]에 근간을 두고 설계된 대화형 사용자 인터페이스를 제공하고 아울러 X-window상에 그래픽 유틸리티를 이용하는 명령어 시뮬레이터이다. 이러한 ISS가 기존의 DLX 시뮬레이터를 근간으로 설계된 이유는 DLX 시뮬레이터가 public 영역에서 이미 검증된 것이고 아울러 새로운 시뮬레이터를 처음부터 설계하는데 걸리는 시간과 노력을 절약할 수 있었기 때문이다. ISS는 C 언어로 구현되여 있고 수행에 따른 그래픽 결과(예, 선긋기 등)는 X-Window상에 display 되도록 설계되여 있다. ISS는 그림 5에서처럼 어셈블리어나 컴파일된 목적 코드를 받아드린 후 이를 그림돌 칩과 똑같이 수행하는 것이다. ISS는 수행결과에 대한 통계적 결과뿐만 아니라 프로그램 수행 중에 debugging 기능까지도 대화형 형태로 제공된다.

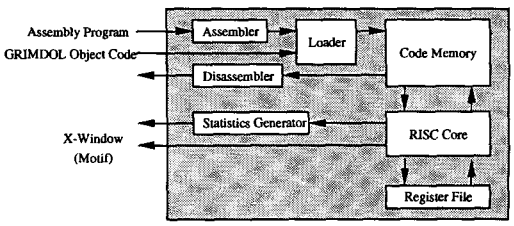

그림 5. ISS : 명령어 시뮬레이터의 구조

그림돌의 C 언어 컴파일러 시스템은 그림 6와 같이 컴파일러, 어셈블러 및 링커와 로더의 세 부분으로 구성이 된다. 즉 header 화일이나 C 소스 화일등을 어셈블리어로 변환시키는 컴파일러와 어셈블리 언어를 binary로 변환시키는 어셈블러, 그리고 여러개의 binary 화일을 하나의 수행 가능한 목적 코드로 묶어주는 링커와 이를 메모리에 로드될 수 있도록 하게 하는 로더가 그것이다. 이때 시스템 측면에서 링크 되어야 하는 라이브러리 화일로는 표준 ANSI C에서 제공되는 표준 라이브러리와 응용 소프트웨어를 위한 그래픽 라이브러리이다. 이 그래픽 라이브러리는 우리의 시스템에서는 Turbo C에서 제공하는 BGI(Borland Graphic Interface) [9] 를 기준으로 하였다. 즉 이 두가지 라이브러리가 모두 binary 형태로 컴파일이 되어 있어야 한다.

지금까지 완성된 ISS와 컴파일러 시스템 그리고 그래픽 라이브러리등을 근간으로 응용 프로그램 시뮬레이터를 구성하여 효율적인 응용 프로그램 개발에 이용하였다. 그림 7은 응용 프로그램 시뮬레이터의 전체 구성을 나타낸다. 이 시스템은 크게 PC(Personal Computer)부분과 WS(Workstation)부분으로 나누어질 수 있다. 먼저 그래픽 interface routine 그리고 그래픽 라이브러리등이 C 언어 컴파일러등을 통해서 binary로 변환된 것을 WS상의 시뮬레이터에 로드를 한다. 그리고 PC 상에서 BGI 라이브러리를 가지고 작성된 응용 프로그램이 수행중에 그래픽 기능을 요하는 BGI routine을 만나면 PC는 PC 내의 communication manager를 통하여 WS의 communication manager쪽으로 어떤 그래픽 routine이 call 되었는지를 알린다. 이러한 통신은 실제로 PC와 WS간의 RS-232C 케이블을 통하여 이루어진다. 이때 요구된 그래픽 routine는 WS상의 ISS 를 통해 수행된 후 그 결과를 display manager module를 통해서 WS상의 X-window에 display한다. 이러한 응용 프로그램 시뮬레이션 환경은 사전에 검증된 그래픽 라이브러리 그리고 이에 따른 컴파일러의 검증등을 통하여 응용 프로그램을 미리 개발해 놓을 수 있는 장점이 있다.

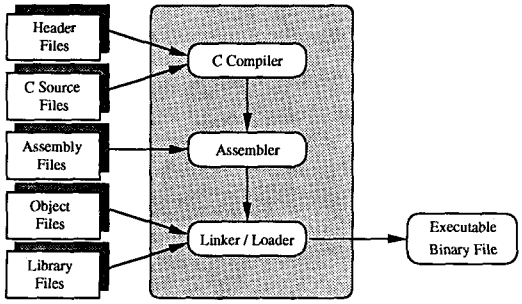

그림 6. 그림돌 컴파일러 구조

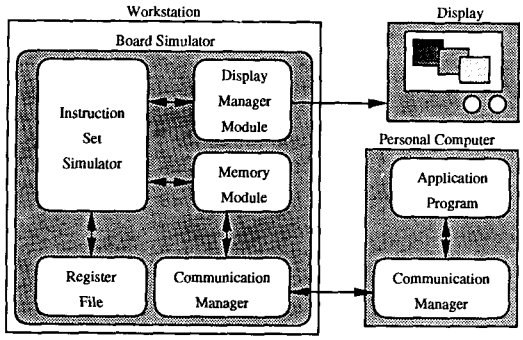

그림 7. 응용 프로그램 시뮬레이터 환경

3. 응용 보드 설계

그림돌 칩이 없는 상태에서 응용 보드를 설계하는 것은 잘못된 설계를 초래할 수 있다. 이에 대한 해결책으로서 Mentor사가 제공하는 BLS(Board Level Simulation)가 있다. BLS는 기본적으로 그림돌의 VHDL 모델과 응용 보드를 구성하는 기타 TTL 라이브러리등이 동시에 시뮬레이션될 수 있는 환경을 의미한다. 그림 8은 그림돌 응용 보드 설계를 위한 전체 환경을 나타내고 있다. 응용 시스템 스펙에 따라 응용 보드를 설계시에 먼저 그림돌 VHDL 모델과 그리고 TTL 과 같은 라이브러리들로 먼저 보드를 구성한다. 이때 PC의 AT 버스라든가 TTL 등으로 구성하기 힘든 부분도 VHDL 모델로 대치되였다. 이렇게 구성된 환경에 몇개의 그래픽 routine(예, 선긋기)등을 수행한 후에 그 결과인 픽셀 데이타를 frame buffer memory의 VHDL 모델이 화일 형태

로 저장한다. 최종 시뮬레이션이 끝난 후 이를 읽어서 스크린으로 모델화된 X-window상에 display함으로써 설계된 응용 보드의 동작을 확인할 수 있다. 이 BLS의 특징은 보드의 기능 검증 뿐만 아니라 정확히 모델링된 타이밍까지도 고려하기 때문에 설계된 칩이 나오기 전에 완벽한 응용 보드를 사전에 설계할 수 있다는 것이고 이것은 동시 공학적인 측면에서 시간과 노력을 절감해 주는 중요한 사항이다.

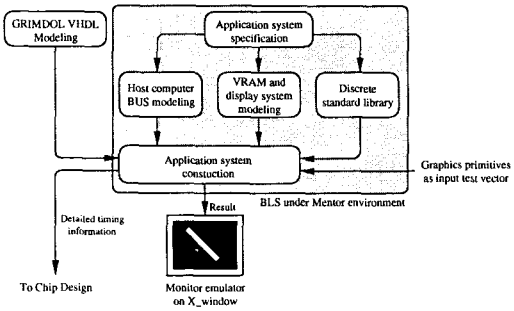

그림 8. BLS(Board Level Simulation) 환경

Ⅳ. 그림돌 및 관련 시스템 측정 결과

제작된 응용 보드는 PC의 AT 버스상에 add-on 카드 형태로 되어있고 스크린으로는 640 x 480의 흑백을 지원하는 보드이다. 칩이 도착한 후 완성된 최종 시스템의 기능 검증을 위한 몇개의 테스트 프로그램을 수행시켜본 바 25MHz에서 잘 수행되는 것을 확인하였다(보드 설계를 수정한 후에는 30MHz 까지 동작함을 확인함). 그러나 막상 완벽한 응용 프로그램을 수행시켜 본 바로는 프로그램의 수행중에 그림돌 칩이 동작을 중지해 버리는 상황이 발생되였다. 이에 대한 테스트를 계속한 결과 칩내의 video system controller가 동작하는 video 모드에서 video system controller가 발생시키는 video memory transfer cycle후에 정상적인 DRAM 억세스를 하지 못한다는 것을 알게 되였다. 이에 대한 설계상의 오류를 추적해 본 바 그림돌의 VHDL 모델에서는 이에 대한 상황을 올바르게 처리하도록 되어 있었으나 로직 설계에서 똑같은 기능이 구현되지 못했기 때문이였다. 이러한 오류의 결정적 원인은 상기의 경우에 대한 시뮬레이션이 하기 어려운 상황에 속했기 때문에 시뮬레이션을 통한 검증을 하지 못했던 것이 실수의 원인이였다. 이에 대한 해결책으로서 우리 설계 그룹에서는 응용 보드에서 video memory transfer cycle 수행된 이후에 무조건 아무런 의미가 없는 호스트 엑세스 사이클을 수행하도록 하였다. 왜냐하면 호스트 사이클이 끝나는 단계에서는 정상적인 DRAM 억세스 사이클 수행으로 올바르게 갈 수 있었기 때문이다. 이와같은 칩의 설계오류를 외부에서 교정한 후에 실제 프로그램은 완벽하게 수행되었다.

그림돌 시스템의 수행 성능을 다른 시스템과의 그래픽 처리속도 비교해 본 결과 PC 386과 PC 486 시스템의 중간이였다. 이는 동시에 같은 응용프로그램을 두 시스템에서 수행시켜 본 상대적 비교 결과에 따른 것이다. 486보다 성능이 늦게 나온 이유는 486 PC에는 데이타 캐쉬가 내장되어 있다는 사실과 우리 시스템이 흑백 시스템이라는 것이 원인이였다고 짐작된다. 왜냐하면 그림돌 칩은 내부에 데이타 캐쉬가 없어서 최소 2 사이클의 메모리 엑세스가 필요하다. 또 그림돌은 8, 16 그리고 32 bit 단위의 픽셀 처리를 잘 하도록 설계되여있기 때문이다. 그림 9과 10는 각각 제작된 응용 보드 및 응용 프로그램이 수행 후 그 결과를 화면에 display한 것을 나타낸다. 그림 9의 응용 보드는 나중에 설계된 full color 보드이다.

상기의 흑백 응용 보드 이외에 하드웨어 그룹에서는 양산성을 고려한 두가지의 응용 보드를 추가로 개발하였다. 그 중 하나는 1024 x 768 화면 resolution을 가지고 full color를 지원하는 스크린용 응용 보드이다. 이 응용 보드의 특징은 기존 VGA(Video Graphic Array)를 이용한 소프트웨어를 위하여 VGA path-through를 가진다. 이 응용 보드를 이용하는 앞으로의 계획은 Windows 가속기인데 디바이스 드라이버등 관련 소프트웨어의 개발 작업은 다음에 언급될 프린터용 보드때문에 지연되고 있다.

양산성을 고려한 또 하나의 응용 보드는 Windows용 프린터 디바이스으로의 응용이다. 현재의 상태는 Windows상의 응용 소프트웨어가 출력한 compressed bitmap 데이타를 받아서 이를 실시간에 decompress 시키면서 이를 LBP(Laser Beam Printer)에 출력할 수 있도록 설계되여 있다. 이 보드의 또 다른 특징은 기존의 300 DPI(Dots Per Inch)의 프린터의 resolution을 600 DPI급 으로 이용될 수 있는 RET (Resolution Enhancement Technique) 기술을 채택하

고 있다는 것이다. 이 일의 궁극적 목적은 bitmap 데이타가 아닌 PostScript 화일을 interpretation 할 수 있는 시스템으로의 발전이다. 현재 이 일의 상태는 관련 하드웨어 및 소프트웨어을 검증중 이며 본 과제의 최종 연도인 내년 후반기에 최종 결과를 예상하고 있다.

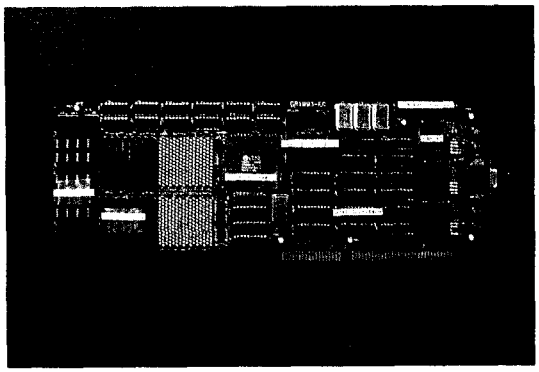

그림 9. 제작된 응용 보드 시스템

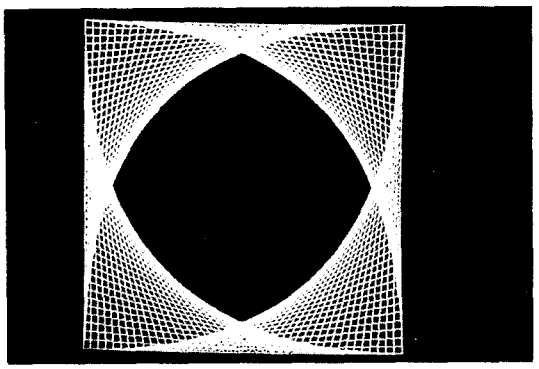

그림 10. 응용 시스템 수행 결과의 화면

V. 결론

지금까지 32 bit RISC 마이크로프로세서인 그림돌 칩 및 관련 시스템인 컴파일러 시스템을 포함하는 시스템 소프트웨어와 ISS를 기본으로 확장 설계된 응용 프로그램 시뮬레이터와 이를 바탕으로 개발된 그래픽 라이브러리를 포함하는 응용 소프트웨어 그리고 BLS (Board Level Simulation)를 통한 응용 보드 설계 등에 대해서 동시 공학의 견지에서 설명하였다. 이러한 동시 공학적인 설계 방법의 결과는 380K 트랜지스트를 가진 그림돌 칩이 30MHz 에서 응용 소프트웨어가 포팅된 응용 보드상에서 성공적으로 동작하게 하였다.

아울러, 그림돌 칩 및 관련 시스템의 설계라는 프로젝트를 수행하면서 몇가지 중요한 사실을 교훈으로 얻었다. 첫째는 동시 공학적인 설계 접근 방법이다. 마이크로프로세서 및 관련 시스템 설계처럼 대규모의 일들은 여러 부분 사람들의 다양한 노력을 필요로 하기 때문에 대규모 일의 효과적인 분할 및 결집이 대단히 중요하다.

둘째는 올바른 설계 방법론이다. 우리 프로젝트의 최장 지연 경로는 그림돌 칩 설계이다. 이는 그림돌 칩의 VHDL 모델을 로직으로 변환하는 과정에서 많은 시간이 걸렸다. 이 부분의 시간을 절약하는 방법으로는 기존의 상위 언어 합성기등을 충분히 이용하는 방향으로 설계 방법론이 정립되어야 하겠다.

셋째는 시뮬레이션 통한 많은 검증이다. 경우에 따라서는 원하는 상황을 시뮬레이션하기가 어려운 경우가 있기 때문에 포기를 해버리는 경우가 왕왕있다. 불가능하지 않는 경우라면 꼭 가능한 모든 경우의 시뮬레이션이 필요하다.

넷째는 하드웨어 prototyping이다. 대규모 시스템을 CAD 소프트웨어로만 시뮬레이션 하는 것은 시간적으로 한계가 있다. 실제로 1 MHz에서라도 동작하는 하드웨어 prototyping이라도 이는 CAD 소프트웨어에서 시뮬레이션하는 경우와 비교가 되지 않을 정도의 속도를 제공한다.

다섯째는 제작된 그림돌 칩의 적극적 활용이다. 왕왕 학교에서 수행된 연구 성과를 단지 연구 수준으로 만족해 버리고 양산이라던가 계속적인 기능 향상을 포기해 버리는 경우가 태반이다. 이는 기술의 계속적 발전을 위해서 바람직하지 않다. 이런 까닭에 우리 설계 그룹에서는 초기 제품의 양산화 그리고 그림돌 칩의 기능 향상을 위해서 계속 노력할 생각이다.

끝으로, 이 논문을 쓰는데 기반이된 연구를 실제로 수행한 사람들은 다음과 같다. 과기원 전산학과 컴퓨터 아키텍춰 연구실의 맹승렬 교수와 길아라, 박승운, 정래훈, 김병호, 유동호, 정의훈, 정재훈, 이은재 등이 컴파일러를 비롯한 소프트웨어 등을 개발하였으며, 전기 및 전자공학과 설계 자동화 연구실의 이윤태, 황규철, 배종홍, 양진혁, 김형원, 박인철, 왕성

문, 홍세경, 성광수 등이 그림돌 칩과 응용 보드 설계를 담당하였다.

參 考 文 獻

[1] Manolis G. H. Katevenis, *Reduced Instruction Set Computer Architectures for VLSI*, The MIT Press, Massachusetts, 1984.

[2] Poul Chow, *The MIPS-X RISC Microprocessor*, Kluwer Academic Publishers, massachusetts, 1989.

[3] Coueignoux P., "Character Generation by Computer", *Computer Graphics and Image Processing*, pp. 240-269, 1981.

[4] Bohm W., Farin G. and Kanmann J., "A Survey of Curve and Surface Method in CAGD", Computer Aided Geometric Design, pp. 1-60, 1984.

[5] *TMS34020 User's Guide*, Texas Instruments, Aug. 1990.

[6] *XL-8220 Processor Preliminary Data Book*, Weitek, Mar. 1990.

[7] *COMPASS Design Navigators Manuals*, COMPASS Design Automation, Inc., 1991.

[8] John L. Hennessy and David A. Patterson, *Computer Architecture a Quantitive Approach*, Morgan Kaufmann Publishers, San Mateo, 1990.

[9] *The BGI Driver Toolkit*, Borland International Corp., 1989.

筆 者 紹 介

慶 宗 旻

1953年 6月 21日生

1975年 2月 서울대학교 전자공학과(학사)

1977年 2月 한국과학기술원 전기 및 전자공학과(석사)

1981年 2月 한국과학기술원 전기 및 전자공학과(박사)

1981年 4月 ~ 1983年 1月 Bell Lab, Murray Hill, NJ/Post doc Member

1983年 2月 ~ 1986年 2月 한국과학기술원 조교수

1985年 1月 ~ 1985年 2月 Univ. of Tokyo, 교환 교수

1986年 5月 ~ 1990年 5月 한국과학기술원 부교수

1989年 2月 ~ 1989年 11月 독일 Karlsruhe 대학/교환교수

1990年 5月 ~ 현재 한국과학기술원 정교수

주관심 분야 : VLSI design, CAD algorithm for VLSI layout and high-level synthesis, Computer graphics algorithm and hardware architectures, Computer systems and microprocessor architectures.