일 반 투 고

C 언어를 사용한 설계 동향
—시스템 및 비메모리 반도체 설계 언어로 부상하는 C 언어—

기 안 도
(주)다이나릿시스템 부설연구소

I. 서 론

1980년대 하드웨어 설계에 주로 사용되었던 EDA(Electronic Design Automation) 소프트웨어들은 Daisy와 Mentor 그리고 Valid 제품들이었고, 이들은 GUI(Graphical User Interface)를 통한 게이트 수준 설계 및 시뮬레이션을 가능케 했다. 1990년대에는 VHDL과 Verilog라는 하드웨어 기술언어(HDL : Hardware Description Language)를 사용한 레지스터 전송 수준 설계가 주를 이루었고, Cadence와 Synopsys가 주도하였다. 최근 들어 HDL보다 상위 수준의 언어로 설계하는 추세가 두드러지고 있다. 본 고에서는 최근 활발한 활동이 진행되고 있는 C 언어에 기초한 설계 동향에 대해 살펴 본다.

II. 배 경

새로운 설계언어가 출현함으로써 설계흐름에 큰 변화를 경험한 적이 있다. 1986년 Gateway Design Automation사에서 Verilog 하드웨어 설계언어를 발표함으로써 게이트 수준(Gate Level)에서 레지스터 전송 수준(Register Transfer Level : RTL)으로 설계수준이 바뀌었다. 이와 유사한 변화가 요즘 다시 나타나기 시작했고 이는 SoC(System-On-a-Chip) 설계가 주류를 이루면서 새로운 설계방법론으로 대두되고 있는 것이다.

현재의 주된 설계흐름에는 여러 언어들이 혼재하는 상황이다. 시스템 규격용 언어로 SDL, 하드웨어 설계언어로 Verilog와 VHDL, 시험용 언어로 Vera, 그리고 내장형 프로그램 언어로 C 등이 그 예라 하겠다. 이러한 여러 언어들을 사용함으로써 각 설계단계를 지날 때 마다 변환과 검증이 필요하며 관리 또한 어렵게 하여 시스템 설계를 복잡하게 하는 요인이 되고 있다.

게이트 수준의 설계에서 RTL 수준의 설계를 거쳐 이제 각종 IP(Intellectual Property)를 이용하는 단계까지 주문형 반도체(ASIC : Application Specific Integrated Circuit) 설계 방법이 발전했다. 이는 ASIC의 집적도가 증가하고 그 구현하고자 하는 기능이 복잡해진데 따른 결과이기도 하지만 한편으로 ASIC 설계도구들의 발전이 이를 가능하게 한 것이다.

이러한 기술적 추이는 기존의 HDL 언어가 RTL 수준을 극복하고 행위모델(Behavioral Model)까지 기능확장을 시도한다거나 PLI(Programming Language Interface) 기능을 통해 시스템 모델링 영역으로 변신을 시도하는 근본적인 이유라고 볼 수 있을 것이다. 그러나 기존의 HDL 언어는 하드웨어 모델에 기초하며 인터프리테이션을 기본으로 한다는 특성으로 인해 고집적/고기능의 SoC 설계에 사용하기에는 여러 가지 어려움이 있다.

내장형 소프트웨어가 비메모리 칩 기능의 50~80%를 차지하는 추세에서 하드웨어와 소프트웨어를 동시에 설계(co-design)하여 같이 시뮬레이션 하는 것(co-simulation)은 피할 수 없는 상황이다. 칩 아키텍터, 시스템 엔지니어, 소프트

웨어 프로그래머가 대부분 C/C++ 언어를 사용하므로 하드웨어 역시 C/C++로 모델링하는 것이 필요하다는 견해가 지배적이다. 칩 자체 뿐 아니라 이를 시험(testbench)하고 사용할 환경(operation environment) 역시 대부분 C/C++로 되는 것이 일반적인 상황이므로, C/C++를 설계의 기본 언어로 고려하는 것은 어쩌면 당연한 일이라 하겠다.

그러나 C/C++ 언어가 하드웨어를 기술하는 데 아래와 같은 것들을 지원하지 못하는 한계가 있다.

- Hardware style communication-signal 등
- Notion of time-Hardware timing-Clock, Delay 등
- Concurrency
- Reactive behavior
- Hardware data types

또한 C/C++ 언어에서 많이 사용하는 것 중 하드웨어로 구현하기 어려운 아래와 같은 것들이 있다.

- Dynamic memory allocation-malloc, free
- Pointers-dereference(*), address-of(&)
- Function calls
- Recursions
- Goto
- Setjmp/longjmp
- Type casting
- Member selection
- Volatile types
- Asm()

이러한 한계를 극복하고자 새로운 구문을 추가하거나, 반대로 사용할 수 있는 구문을 제한하거나, 또는 하드웨어 모델링용 라이브러리를 추가하는 등의 방법이 사용되고 있다.

C 언어를 사용하여 하드웨어를 모델링할 경우 다음과 같은 장점을 기대할 수 있다.

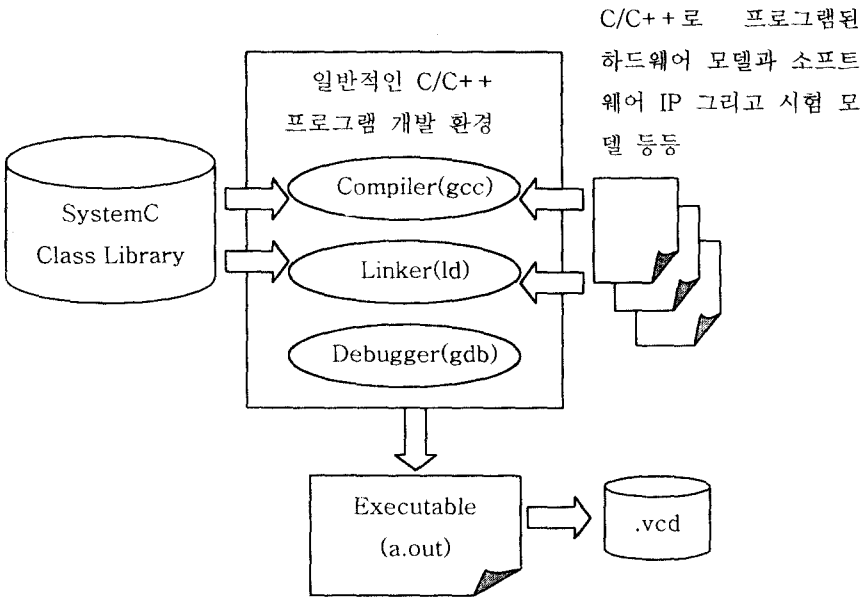
- 소프트웨어와 동일한 환경을 사용하므로 시스템 전체 모델링이 용이해진다.
- 기존의 C 프로그램 개발 환경을 그대로 활용할 수 있다.
- 시뮬레이션이 빨라져 기능 검증 시간이 단축된다.
- 개발된 모델을 '실행가능 규격(executable specification)'으로 사용할 수 있다.
- 컴파일러 또는 변환 프로그램을 이용하여 바로 합성 가능한 HDL을 얻을 수 있다.
- C 수준 에뮬레이션 기술 및 도구를 이용하여 고속/실시간 에뮬레이션이 가능하다.
- 하드웨어에 덜 적응된 일반 소프트웨어 개발자도 하드웨어를 모델링하여 개발할 수 있다.

여러 회사와 조직들이 나름의 C 언어 확장이나 라이브러리를 개발하고 있어 호환성 문제 그리고 아직 C 언어로 된 IP들이 다양하지 못하다는 단점은 있으나 이러한 부분을 해결하고자 국제적인 움직임이 활발하게 전개되고 있다. SpecC와 SystemC가 대표적인 것으로 이들은 하드웨어를 알고리즘부터 레지스터 전송 수준까지 모델링할 수 있도록 한다.

Ⅲ. SpecC와 SystemC

1996년 UCI(University of California, Irvine; www.ics.uci.edu/~specc)에서 Toshiba와 Hitachi 등의 후원으로 SpecC(Specification description language based on C) 개발이 시작되어 1997년에 Release 1.0, 1998년에 Release 2.0이 발표되었다. 1999년 11월에는 STOC(SpecC Technology Open Consortium ; www.specc.org)가 결성되어 이를 중심으로 활동하고 있다. STOC에는 2000년 5월 현재 Cadence, CynApps, Mentor 등을 포함한 여러 회사들이 참여하고 있다.

SpecC로 작성된 설계는 전용 컴파일러를 사용하는데, 내부적으로 C++ 파일을 만들고 이것을 GNU gcc와 같은 범용 컴파일러로 컴파일하여



〈그림 1〉 SystemC 개발 환경

실행 가능한 결과를 생성한다. 이 결과를 실행시킴으로써 모델된 설계를 시뮬레이션하게 된다.

1999년 9월에 OSCI(Open SystemC Initiative ; www.systemc.org)라는 단체가 결성과 동시에 SystemC Ver0.90을 발표했다. 이후 2000년 3월에 Ver1.0, 2001년 10월에 Ver2.0을 발표하였다. OSCI에는 Synopsys, CoWare, Frontier, Cadence, Mentor를 포함한 여러 회사들이 참여하고 있다.

SystemC는 〈그림 1〉에서 보는 바와 같이 C^{++} 라이브러리 클래스만을 추가함으로써 시스템을 모델링할 수 있게 하고, 범용 컴파일러를 그대로 사용한다. 따라서 시스템 모델링과 디버깅 환경을 따로 갖출 필요없이 기존의 C/C^{++} 프로그램 개발 환경만으로 시뮬레이션이 가능하다.

STOC와 OSCI가 SpecC와 SystemC를 EDA 산업계 표준으로 자리잡을 수 있도록 노력하는 것 이외에 C/C^{++}를 근간으로 독자적인 설계 및 시뮬레이션 환경을 만들고, 더 나아가 합성도구를 개발하고 판매하는 회사들도 있다. 대표적으로 Forte의 Cyn^{++}/Cynchor, CoWare의 N2C, Co-Design의 SystemSim, C-Level Design의 CSIM, Adelante의 A|RT 등이 있다. 이들 회사들은 C/C^{++} 뿐 아니라 Verilog와 VHDL 등도 시스템 모델링에 사용할 수 있도록 하는 환경을 제공하기도 하며, 이중 Co-Design의 Suprerlog 언어는 C/C^{++}, Verilog, VHDL로 변화하거나 이들로부터 변환되는 환경을 제공하고, Forte는 SystemC와 유사한 접근 방법을 취하며, Verilog로 변환하거나 co-simulation을 지원한다.

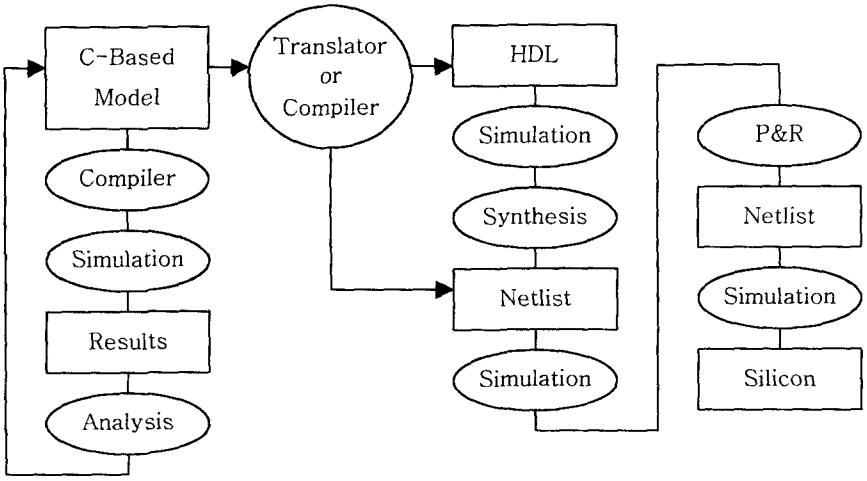
현재까지 알려진 회사들을 정리하면 아래와 같다.

- Adelante Technologies(Frontier Design) — http://www.frontiered.com ; A|RT ; Architectural synthesis using C or SystemC.
- Celoxica — http://www.celoxica.com ; Handel-C ; C-based design to FPGA.
- C-Level Design — http://www.cleveldesign.com ; CSIM 2.0 ; Comprehensive simulation environment for C/C^{++} class library-based design and verification.
- Co-Design Automation — http://www.co-

design.com ; SystemSim ; Multilingual simulator, supporting Verilog, Superlog, C, C++ and SystemC without interfaces or co-simulation.
- CoWare—http://www.CoWare.com ; N2 C ; Full SystemC co-design environment.
- Cselt—http://www.cselt.it ; Vip Library; A wide set of customizable and flexible system level IP Soft Cores.
- Cynergy System Design—http://www.cynergysd.com ; AfterBurner, ArchGen, ASVP Builder
- Dynalith Systems—http://www.dynalith.com ; iSAVE, in-system verification engine
- Forte(CynApps) — http://www.cynapps.com ; Cynlib
- Future Design Automation—http://www.future-da.co.jp ; Design Prototyper ; Architectural synthesis tool.
- Inter Design Technology—http://www.interdesign.co.jp VisualSpec SpecC based specificaton authoring tool.
- Synopsys — http://www.synopsys.com ; CoCentric SystemC Compiler ; Synthesis of hardware from SystemC models.
- Tenison Technology—http://www.tenisontech.com ; VTOC, CTOV
- Veritools — http://www.veritools.com ; SuperC ; A very fast SystemC simulator that writes a highly compressed data format.
- Virtio — http://www.virtio.com ; Prototyping to SystemC.
- Y Explorations—http://www.yxi.com ; eXCite ; The C based synthesis solution.

C 언어에 기초하여 시스템을 모델링하는 경우 〈그림 2〉와 같은 설계흐름을 생각해 볼 수 있다. 즉 C/C++로 시스템을 모델링하며, 이때 모델링되는 것 외에 소프트웨어를 포함한 주변의 모든 것들 역시 C/C++로 모델링 또는 프로그램한 후, 컴파일하여 시뮬레이션 한다. 그 결과가 만족스러우면 C/C++ 코드를 하드웨어 설계 언어로 자동 변환하거나 또는 곧바로 합성 가능한 코드를 생성한 후 기존의 설계 흐름으로 ASIC을 개발할 수 있다.

C/C++에서 합성 가능한 HDL을 생성하는 도구로 2000년 6월 미국 LA에서 개최된 제37차 설계자동화학회(Design Automation Conference)에서 Synopsys가 발표한 CoCentric Sys-



〈그림 2〉 설계 흐름도

temC Compiler라는 소프트웨어 제품이 있고, 이외에 Adelante, CoWare, Forte, C-Level Design 등이 이러한 도구를 개발하여 발표하였거나 개발중으로 알려지고 있다. 또한 동학회의 전시회에서 Dynalith Systems(www.dynalith.com)가 C 언어로 된 설계를 하드웨어로 에뮬레이션 하는 iSAVE 제품을 발표 및 전시함으로써 C 언어 설계의 중요한 전기를 마련하였다. 특히 iSAVE는 Quickturn 제품이나 IKOS 제품과 같은 네트리스트를 필요로 하는 기존의 하드웨어 에뮬레이터와는 근본적으로 다른 C 수준 모델을 하드웨어로 에뮬레이션하는 기술을 이용하여 100만 게이트 크기에 맞먹는 ASIC을 행위모델 수준에서 거의 실시간으로 에뮬레이션할 수 있음을 보였다. 따라서 C 언어로 설계된 모델을 시뮬레이션하거나, 에뮬레이션하거나, 합성 가능한 코드를 얻는 등 모든 설계 흐름이 정립되었다.

Ⅳ. 종 합

반도체 공정의 발전으로 집적도가 증가하여 디지털 설계의 중요한 전기를 마련하였고, 고집적 반도체 기술은 보다 복잡한 시스템을 반도체로 만들 수 있게 했으며, 이제 하드웨어와 소프트웨어가 혼재하는 구성을 갖게 되었다. 따라서 EDA 도구들은 하드웨어와 소프트웨어를 동시에 지원할 수 있어야 했는데, 이런 부분에서 설계언어가 가장 심각한 문제라 할 수 있다. 대부분의 내장형 소프트웨어는 C 언어로 개발되고 있고, 시스템 설계자들은 시스템 수준의 모델링에 C/C++ 언어를 주로 사용하여 시스템의 성능을 예측하고 기능을 검증하는 반면 실제 설계에는 합성 가능한 하드웨어 기술언어를 사용하였다. 단일 언어로 설계할 경우 하드웨어와 소프트웨어를 동시에 시뮬레이션 하기도 용이해질 수 있기 때문에 자연적으로 하드웨어와 소프트웨어를 모두 기술할 수 있는 단일 언어의 필요성이 대두되었다. 이러한 추세는 C 언어를 가장 강력한 후보로 부상시켰으며 이러한 배경에 STOC와 OSCI가 SpecC와 SystemC를 업계표준으로 만들려는 노력을 하고 있는 것이다. 향후 SoC 설계 기법이 비메모리 반도체 설계의 주요 설계 흐름이 될 것이라는 전망은 이견의 여지가 없고, 이러한 추세에서 C 언어에 기초한 하드웨어 모델링 방법론 역시 중요한 기술로 자리 잡게 될 전망이다.

저 자 소 개

奇 安 度

1997년 11월 University of Manchester, Department of Computer Science, Ph. D., 1988년 2월 KAIST, 전기 및 전자공학과, 석사, 1986년 1월 한양대학교, 전자공학과, 학사, 1989 년 2 월 ~1989 년 7 월 : University of Washington, Visiting Scholar, 1991년 9월~1991년 12월 : 서강대학교, 시간강사, 1988년 3월~2000년 6월 : 한국전자통신연구원, 선임연구원, 2000년 6월~현재 : (주)다이나릿시스템, 책임연구원, 〈주관심 분야 : 컴퓨터구조, 병렬처리, In-system verification〉