특 집

SoC Front-end 설계를 위한 통합 환경

김 기 선, 김 성 식, 이 희 연, 김 기 현, 채 재 호
하이닉스 반도체 주식회사/System IC 컴퍼니

Abstract

In this paper, we introduce an integrated SoC front-end design & verification environment which can be practically used in the embedded 32-bit processor-core SoC VLSI design. Our introduced SoC design & verification environment integrates two most important flows, such as the RTL power estimation and code coverage analysis, with the functional verification (chip validation) flow which is used in the conventional simulation-based design. For this, we developed two simulation-based in-house tools, RTL power estimator and code coverage analyzer, and used them to adopt them to our RTL design and to increase the design quality of that. Our integrated design environment also includes basic design and verification flows such as the gate-level functional verification with back annotation information and test vector capture & replay environment.

I. 서 론

미세 공정의 발전에 따른 집적도의 향상은 동일한 면적에 집적할 수 있는 게이트 수(Gate count)의 엄청난 증가가 가능하도록 만들었으며, 이에 따라 예전엔 상상하지도 못할 정도로 다양한 기능을 한 칩(Chip)내에 집적시킬 수 있는 SoC(System On a Chip)로의 구현 또한 가능해졌다. 공정의 미세화는 반도체 생산에 있어서 예전과는 달리 많은 비용이 소요되도록 만들었으며, 생산에 걸리는 시간 역시 다양하고 복잡한 공정을 반영하여 상당한 수준까지 증가되고 있다.

반도체 설계와 생산에서의 겪게 되는 이와 같은 비용/시간 측면에서의 어려움과 함께, 반도체 시장 환경 역시 제품(Device)의 개발 단계부터 시장에 제품이 출시되기까지의 시간(Time-to-market)이 점점 짧아지고 있으며 우선 출시에 따른 시장에서의 선점 효과의 정도 역시 예전과는 달리 매우 커지고 있다. 결국 반도체, 특히 비메모리(Non Memory) 반도체와 관련된 설계/생산/마케팅 분야는 점점 더 어려운 상황과 현실에 직면하고 있다고 할 수 있다. 이에 따라 반도체의 개발 단계, 특히 설계 단계에서 향후 발생할 수 있는 다양한 오류들로 인한 설계의 지연과 수정, 시간과 비용의 추가적인 투입 및 시장에서의 선점적 지위의 상실 등을 미리 효과적으로 방지하기 위한 노력을 기울이는 것이 설계자가 가져야 할 가장 중요한 책무 가운데 하나가 되었다.

이를 위하여 설계의 각 단계에서 다양한 검증 방법을 통하여 현재 진행하고 있는 설계 단계에서 발생하고 있는 문제점들을 미리 파악하고, 이 정보를 바탕으로 설계의 다음 단계에서 발견될 수 있는 문제점들에 대한 조치를 미리 취할 수 있다면, 설계 단계에서 발생할 수 있는 지연의 문제를 어느 정도 방지할 수 있을 것이다. 이는 요

즘의 반도체 설계, 특히 SoC와 같이 다양한 기능을 수행하는 많은 블록(Block)들이 한 칩 내에 집적되는 상황에서 매우 중요한 의미를 갖게 되었다. 결국 SoC를 위한 디지털(Digital) 회로 설계의 맨 첫 단계라 할 수 있는 레지스터 전송 수준(RTL : Register Transfer Level)에서 다양한 방법을 이용한 사전 검증(Beforehand Verification)은 반드시 필요한 필수 설계 흐름(Design flow)의 하나가 되었다.

하지만 레지스터 전송 수준 단계에서 다양한 방법/수단을 이용하여 검증 작업을 진행하는 것은 기존의 동작 검증(Functional Verification)에 익숙한, 즉 시뮬레이션에 기반(Simulation Based)한 설계 환경에 익숙해진 설계자로 하여금 새로운 방법과 수단에 익숙해 지기 위하여 추가적인 노력과 시간을 투입하게 만드는 등, 또 다른 부담으로 작용하게 되는 단점이 있었다. 저전력 설계가 하나의 필수(Design Necessity)가 되어버린 지금의 현실에서 전통적인 설계 흐름(Conventional Design Flow)에 익숙한 설계자들은 합성(Synthesis) 과정을 거친 게이트 수준(Gate Level)의 구현에서나 전력 소모를 예측을 기대하고 있다. 또한 레지스터 전송 수준에서의 동작 검증에 사용한 테스트 코드 또는 벤치(Test Code or Bench)의 충실 도가 어느 정도 되는지에 대한 계량적인 분석(Quantitative Analysis) 결과 없이 게이트 수준(Gate Level)의 구현에서나 결함(Fault) 분석 정도를 진행해 왔던 것이 지금까지의 설계 현실이었다.

레지스터 전송 수준의 설계는 디지털 반도체 설계의 맨 처음 출발점이며, 이 수준에서의 설계는 뒤의 게이트 수준의 설계, P&R(Placement & Route) 단계의 설계에 까지 절대적인 영향을 미치게 되고, 궁극적으로 생산(Fabrication)하여 시장(Market)에 출시될 반도체의 전체적인 동작과 특성을 좌우하게 된다. 미리 계획된 칩의 올바른 동작을 보장(Guarantee)하기 위하여, 레지스터 전송 수준의 설계에서 테스트 코드의 적용 범위(Coverage)를 분석하고 전력 소모 특성을 미리 알아보기 위한 레지스터 전송 수준에서 전력 소모 예측(Power Consumption Estimation)등의 작업을 진행하는 것은 매우 중요한 의미를 갖는다. 이 두 가지 항목을 위한 여러 반도체 설계 툴(Design Tool)들이 개발되어 사용되고 있으나, 동작 검증에 기반한 기존의 설계 환경과는 잘 맞지 않아 이를 적용하지 위하여 추가적인 시간과 노력을 투자해야 하는 경우가 많았다.

위와 같은 문제점을 해결하기 위하여, 기존의 시뮬레이션에 기반한 동작 검증 환경(Simulation Based Functional Verification Environment)에 익숙한 설계자들이 동작 검증에 사용한 설계 환경을 그대로 이용하여 레지스터 전송 수준에서 전력 소모를 예측하고 코드의 적용 범위를 분석할 수 있는 자체 툴(in-house tool)을 개발 하였으며 이를 기존의 SoC Front-end 설계 환경에 통합하여 효율적인 설계/검증이 가능하도록 하였다.

본 논문에서는 SoC의 Front-end 통합 설계 환경에 이용되도록 시뮬레이션 결과에 기반하여 동작하도록 개발한 레지스터 전송 수준의 전력 소모 예측 프로그램(RTL Power Estimator)과 코드의 적용 범위 분석 프로그램(Code Coverage Analyzer)을 소개하고 이 두 툴이 기존의 전통적인 SoC Front-end 설계/검증 환경에 어떻게 적용되어 사용되는지를 소개한다.

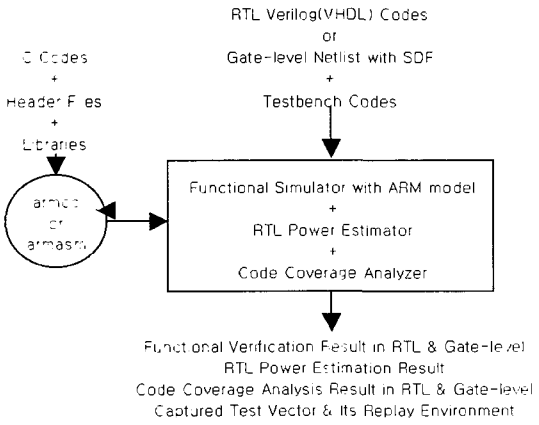
본 논문은 다음과 같이 구성되어 있다. II장에서는 SoC Front-end 설계를 위한 통합 환경, III장에서는 II장에 소개된 레지스터 전송 수준에서의 전력 소모 예측 방법을, 그리고 IV장에서는 코드의 적용 범위 분석 방법에 대하여 소개한다. V장에서는 기존의 전통적인 설계 검증 방법인 시뮬레이션과 본 논문에서 소개된 두 툴을 함께 사용하였을 때의 성능 비교 결과를 보여주며 마지막으로 VI장에서 결론을 짓는다.

II. SoC Front-end 통합 설계 환경

〈그림 1〉은 시뮬레이션에 기반한 SoC의 Front-

end VLSI 설계를 위한 통합 환경의 전체적인 동작 흐름을 보여준다.

본 환경은 32-bit 프로세서 코어 내장형(Embedded 32-bit Processor Core) SoC를 개발할 때 이용되는 칩 검증(Chip Validation)환경으로, 전통적인 동작 검증 환경인 시뮬레이션을 기반으로 하여 기존의 전통적인 설계/검증 환경(Conventional Design & Verification Environment)에 익숙한 설계자들이 추가적인 노력과 시간의 투자가 없이도 쉽게 사용할 수 있도록 되어 있다. 32-bit 프로세서 코어 내장형 SoC를 개발할 때 기본적으로 이용하는 컴파일러(armcc) 또는 어셈블러(armasm)를 사용하여, 검증에 이용할 소스 코드(Source Code : C Codes, Header Files, Libraries)들을 가지고 레지스터 전송 수준에서의 동작 검증을 수행하는 것을 기본으로 하여, Back-annotation 정보를 이용한 게이트 수준의 검증에도 이용할 수 있으며, 칩 제조 이후의 테스트 단계에서 이용할 수 있도록 테스트 벡터(Test Vector)를 생성하는 기본적인 기능을 가지고 있다. 이 외에 칩의 올바른 동작을 보장하기 위한 코드의 적용 범위 분석과 전력 소모 특성을 미리 알아보기 위한 레지스터 전송 수준에서의 전력 소모 예측 기능도 함께 포함하고 있다.



〈그림 1〉 SoC Front-end 통합 설계/검증 환경

〈**Fig. 1**〉 Integrated SoC Design & Verification Environment

앞의 제I장 서론에서 잠시 언급한 바 있듯이, 레지스터 전송 수준에서 행하여질 수 있는 시뮬레이션 결과에 기반한 여러 종류의 분석들(Analysis) 가운데 전력 소모 예측과 코드의 적용 범위 분석은 점차로 중요한 설계 이슈(Design Issue)가 되고 있다. 지금까지의 전통적인 설계에서 전력 소모는 게이트 수준 이하의 단계에서 주로 논의 된 항목이었으나, 설계에 사용되는 게이트 수의 엄청난 증가와 기능의 복잡/다양화는 게이트 수준에서의 전력 소모 예측 단계를 레지스터 전송 수준까지 끌어 올려야 하는 상황에 이르게 되었다. 레지스터 전송 수준의 설계 단계에서 미리 전체 chip의 전력 소모를 예측하여 적절한 대응을 하지 않으면, 로직 합성(Logic Synthesis) 단계 이후에서야 전력 소모의 문제점을 발견하고 다시 레지스터 전송 수준으로 돌아와 문제를 해결해야 하는 등, 재설계 또는 설계 수정에 따른 시간과 비용 측면에서의 매우 많은 손해를 예상할 수 있다.

코드의 적용 범위 분석 역시 레지스터 전송 수준에서의 설계/검증에 중요한 의미를 갖는 항목 중의 하나이다. 칩의 검증에 사용된 테스트 코드(Test 코드 : C Codes or Assembler Codes)가 얼마나 충실하게 설계된 칩 전체를 검증할 수 있는지를 계량적으로 분석하는 항목으로, 보통 레지스터 전송 수준의 검증에 사용된 테스트 벡터가 게이트 수준의 검증뿐만이 아니라 칩 테스트 단계에도 직접 적용되고 있음을 감안해 볼 때, 레지스터 전송 단계에서 미리 코드의 적용 범위 분석을 하게 되면 그만큼 충실한 테스트 코드를 만들 수 있으며, 이를 이용하여 높은 커버리지(Coverage)를 갖는 테스트 벡터, 특히 동작 검증에 이용할 수 있는 벡터의 생성이 가능하다.

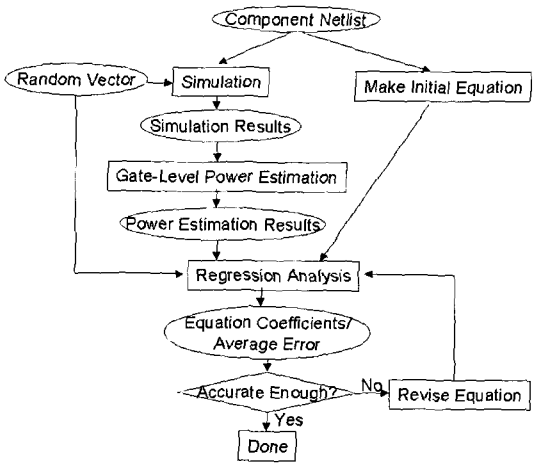
본 논문은 기존의 전통적인 동작 검증 중심의 설계 환경, 즉 시뮬레이션에 기반하여 레지스터 전송 수준의 전력 소모 예측과 코드의 적용 범위 분석을 동시에 수행할 수 있도록 통합된 SoC Front-end 설계 환경을 소개한다.

Ⅲ. RTL Power Estimation

본 논문에서 소개하는 SoC Front-end 통합 설계 환경의 가장 큰 핵심 부분은 시뮬레이션을 진행하는 동안 레지스터 전송 단계에서의 전력 소모를 미리 예측할 수 있는 수단을 제공한다는 점이다. 기존에 주로 이용되었던 방법인 대략적인 스위칭 정도(Switching Activity)만을 이용한 전력 소모 예측 방법과는 달리, 동작 검증에 직접 이용된 테스트 코드를 이용하여 시뮬레이션 결과에 기반한 동적인 분석(Dynamic Analysis)이 가능함에 따라 테스트 코드가 가지고 있는 데이터 상관관계(Data Correlation)등에 의한 전력 소모를 매우 사실적으로 예측할 수 있는 장점이 있다. 본 설계 환경에 통합되어 있는 레지스터 전송 수준에서의 전력 소모 예측 프로그램은 회로의 형태에 따른(Circuit Dependent) 독특한 회귀 모델(Regression Model)을 사용하여 빠른 전력 소모 예측이 가능하도록 해 주며, 순수한 동작 검증의 경우와 비교해 볼 때 그리 크지 않은 시간적 오버헤드(Overhead)를 나타낸다.

기존에 레지스터 전송 수준의 전력 소모 예측에 주로 이용되어 왔던 수작업으로 각 구성 요소(Component)별 파워 모델(Power Model)을 만드는 방법은 정확하기는 하나 특정 하드웨어(Hardware)에 지나치게 의존하게 되는 단점이 있었다. 그리고 파워 모델을 준비하고 만드는 과정에서 많은 수작업(Human Effort)을 요구하게 됨에 따라 일반적으로 널리 사용되는 데에는 많은 무리가 있었다[3, 4]. 이 문제를 해결하기 위하여 자동으로 파워 모델을 생성하는 여러 방법들[5, 6, 7, 8]이 제안되었다. 본 논문에서 구현한 방법은 [5]와 비슷한 방법이나, 좀 더 규칙적으로 구성된 회로(Regularly Structured Circuit)에 쉽게 적용할 수 있는 방법이다.

레지스터 전송 수준의 전력 소모 예측에서는, 해당 설계를 가지고 구현(Implementation)하게 될 정확한 목표 모델(Target Model)을 알아내기가 어렵기 때문에 소모되는 전력 값을 정확히 예측해내는 것이 매우 어렵다 따라서 정확한 예측이 목적이 아니라, 전체 칩에서 보이는 전력 소모의 경향(Tendency)을 알고자 하는 것이 주된 이용 목적이라고 할 수 있다.



〈그림 2〉 각 구성 요소 별 파워 모델을 만들기 위한 자동화 절차

〈Fig. 2〉 Automated Progress of deriving a component power model equation

〈그림 2〉는 본 논문에서 사용하는 각 구성 요소 별 파워 모델을 만드는 자동화된(Automated) 절차를 나타낸 것이다. 각 구성 요소는 다음과 같은 형태의 파워 모델을 갖는다.

$$E_{comp(n)}=\sum_i c_i x_i$$

여기에서 comp(n) 워드 길이(Word Length) 또는 회로 크기(Circuit Size) n을 갖는 각 구성 요소를 나타내며, c_i는 실제로 소모되는 전력 값을 이용한 회귀 분석(Regression Analysis)을 통하여 구하여진 모델 계수(Model Coefficient), 그리고 x_i는 각 회로에 관련된(*Circuit Dependent*) 파워 파라미터(Power Parameter)이다. 파워 파라미터는 다음과 같은 형태를 갖는다.

$$x_i=f(\mathbf{v}(t-1),\ \mathbf{v}(t))$$

여기에서 v(t-1)와 v(t)는 각 구성 요소로 들어오는 연속적인 입력 벡터의 일부(Subsequent

Input Vector)이다. 파워 파라미터는 입력으로 들어오는 각 벡터를 기준으로 회로의 구성과 내부 노드(Node)에서의 스위칭 조건(Switching Condition), 글리치(Glitch) 생성(Generation) 및 전파(Propagation), 그리고 입력/출력 핀에서의 정전 용량(Capacitance)등을 고려하여 정하여진다.

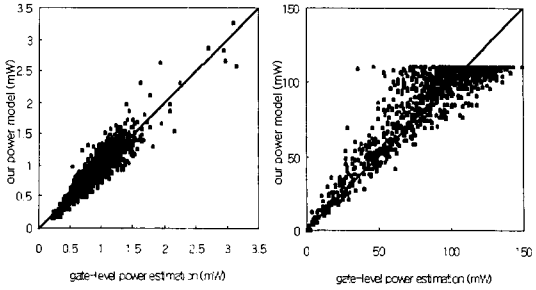
특정한 크기를 갖는 구성 요소에 대한 모델은 다음과 같은 일반적은 형식(General Form)으로 통합될 수 있으며 여기에서 N은 회로 크기(Circuit Size)로 N-bit으로 구성된 구성 요소를 나타낸다.

$$E_{comp}=\sum_i c_i x_i+N\sum_j c_j x_j+N^2\sum_k c_k x_k+\cdots$$

〈그림 2〉의 절차를 이용하여 자동으로 각 구성 요소 별로 c_l의 값을 구하여 레지스터 전송 수준의 전력 예측에서 이용할 파워 라이브러리(RTL Power Library)를 만든다.

각 구성 요소는 디지털 로직(Digital Logic)이 이용하는 가장 기본적인 구성 단위인 레지스터, 멀티플렉서(Multiplexer), 덧셈기(RCA : Ripple Carry Adder), 곱셈기(AM : Array Multiplier or Pipelined AM), 비교기(Equality Comparator), 뺄셈기(RBS : Ripple Borrow Subtracter), 그리고 기본적인 로직 유닛(Logic Unit : AND, OR, NAND, NOR, INV, XOR, XNOR 등) 들로 이루어져 있다. 특히 연산(Arithmetic Operation)에 관련된 파워 파라미터는 경험에 의한 방법(Rule of Thumb)을 사용하여 정의하였으며, 디지털 로직의 기본 구성 단위들과는 다른 좀 더 복잡한 모델링 과정을 거쳐서 생성하였다. 자세한 방법은 [9]에 소개되었다.

〈그림 3〉은 본 논문에서 제안한 방법을 적용하여 32bit RCA와 32bit AM의 레지스터 전송 수준의 전력 소모 모델이 나타내는 전력 소모 예측과 게이트 레벨 구현에서의 전력 소모 예측 결과[10]의 차이를 보여준다. 레지스터 전송 수준에서의 전력 소모 예측 결과와 게이트 수준에서의



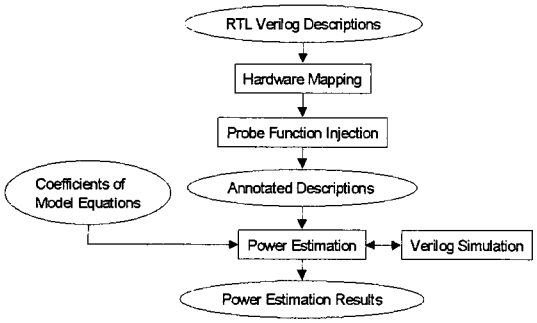
(a) 32bit RCA (b) 32bit AM

〈그림 3〉 게이트 수준에서의 실제 전력 소모와 레지스터 전송 수준에서의 전력 소모 예측 비교

〈**Fig. 3**〉 Comparison between gate-level power consumption and RTL power estimation

전력 소모가 상당한 수준의 상관 관계를 갖고 있음을 보여준다.

디지털 로직 외에, 메모리(Memory)나 프로세서 코어(Processor Core), ADC(Analog to Digital Converter), DAC(Digital to Analog Converter), 그리고 PLL(Phase Locked Loop)과 같은 매크로(Macro)들은 각자의 전력 소모 모델, 또는 실제 전력 소모 값을 이용하여 구성된 자료를 바탕으로 비슷한 방법의 회귀 분석으로 통하여 전력 소모 모델 구성 및 파워 라이브러리 구성을 하게 된다. 현재 하이닉스 반도체의 0.6 um/0.5 um/0.35 um/0.25 um/0.18 um Standard Cell Library에 대하여 파워 라이브러리가 준비되어 있으며, 이렇게 만들어진 레지스터 전송 수준의 파워 라이브러리를 이용하여



〈그림 4〉 레지스터 전송 수준의 전력 예측 흐름

〈**Fig. 4**〉 RTL Power Estimation Flow

〈표 1〉 파워 모델링 결과

〈**Table 1**〉 Power Modeling Results

Module	Size	# Coef.	Error	Module	Size	# Coef.	Error
RCA	8	2	19.6%	AM	8	4	19.3%
	16		16.3%		16		16.5%
	32		12.7%		32		16.0%
RBS	8	2	18.6%	3 Stage PipeAM	8	4	4.3%
	16		13.6%		16		7.9%
	32		11.9%		32		9.7%
E.Comp	8	3	7.1%	4 Stage PipeAM	8	4	4.6%
	16		5.3%		16		5.7%
	32		8.9%		32		7.6%
M.Comp	8	2	19.7%	MUX Tree	4	3	9.2%
	16		12.7%		8		8.1%
	32		12.9%		16		9.7%
Register	N	3	0.0%	Logic Unit	N	2	0.8%

〈표 2〉 전력 소모 예측 결과

〈**Table 2**〉 Power Estimation Results

Circuit	#Gates	Gate-level	RTL	Error
GCD	283	7.695mW	7.557mW	-1.8%
AM2901	1549	13.045mW	12.533mW	-3.9%
RS-ECC	20882	71.472mW	58.348mW	-18.4%
LCD Driver	14238	1.403mW	1.107mW	-21.1%
DSP	87393	24.746mW	22.705mW	-8.2%

다음과 같은 방법으로 레지스터 전송 수준에서의 전력 소모 예측을 수행한다. 〈그림 4〉에 자세한 흐름이 나타나 있다.

우선 합성이 가능한 하드웨어 기술 언어(HDL : Hardware Description Language, Verilog or VHDL)로 작성된 레지스터 전송 수준의 소스 코드(Source Code : Test-bench 제외)가 들어오게 되면 이를 파싱(Parsing)하여 각 레지스터 전송 수준의 기술(Description)들을 레지스터 전송 수준의 파워 라이브러리가 가지고 있는 각 구성 요소로 하드웨어 할당(Hardware Mapping)을 한다. 이 작업은 실제 레지스터 전송 수준의 코드 상에 PI(Procedural Interface)의 한 종류인 PLI(Programming Language Interface) 함수를 덧붙이는 작업(Annotation)으로, 각 구성 요소 별 PLI 함수는 회귀 분석으로 통하여 만들어진 파워 모델을 내장하고 있으며 PLI 함수들을 덧붙인 레지스터 전송 수준의 코드를 이용한 동작 검증이 진행되는 동안 이미 구축된 파워 라이브러리를 사용하여 RTL 전력 소모를 예측을 동시에 진행하게 된다. 〈표 1〉, 〈표 2〉는 하이닉스의 0.25um, 2.5V standard cell library를 사용한 파워 모델링 결과와 이를 적용한 레지스터 전송 수준에서의 전력 소모 예측 결과를 보여준다.

〈표 2〉의 결과를 보면, 게이트 수준에서의 전력 소모에 비하여 레지스터 전송 수준에서의 전력 소모가 더 적은 값을 보임을 알 수 있다. 게이트 수준에서는 클럭 트리 합성(CTS : Clock Tree Synthesis)에 의하여 형성된 클럭 트리와, 합성된 회로 내에서 발생하는 글리치(Glitch) 등에 의한 추가적인 전력 소모 요인이 있다. 〈표

2>에서 #Gates는 2-input NAND Gate로 환산한 게이트 수로 다양한 하드 매크로(Hard-macro : compiled memory, ADC, DAC, PLL, ARM core 등)들을 제외한 순수 레지스터 전송 수준에서 사용된 코드만을 고려한 값이다.

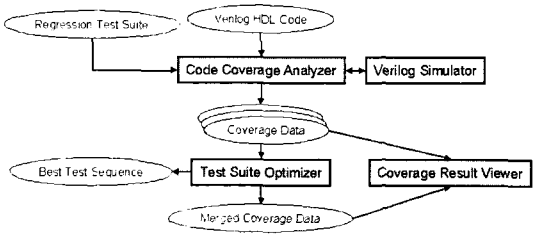
일반적으로 레지스터 전송 수준의 전력 소모 예측은 다음과 같은 이유로 인하여 구현(Gate-level 또는 P&R) 이후와 상이한 결과를 보이게 되며 정확도 자체보다는 전력 소모 경향(Tendency) 예측에 더 큰 의미를 갖는다.

- 파워 모델 자체가 완벽하지 못하다.
- 합성(Synthesis) 과정을 거치는 동안 대부분의 로직은 최적화(Optimization) 과정을 거치게 된다.
- 로직 합성/클럭 트리 합성 중간에 상당한 양의 버퍼(Buffer)들이 삽입된다.
- 게이트 수준에서의 전력 소모 예측에서는 입력으로 들어오는 신호들과 내부 노드(Node)들에서의 신호의 기울기(Slope)을 고려하나 레지스터 전송 수준에서는 이를 고려하기가 어렵다.
- 게이트 수준에서는 많은 양의 글리치(Glitch)가 발생할 수 있으나 레지스터 전송 수준에서는 이의 고려가 어렵다.
- 파워 라이브러리는 최소 구동 사이즈(Driving Size)를 갖는 게이트들을 이용하여 만들었기 때문에, 실제 게이트 수준 구현에서의 다양한 구동 사이즈를 반영하기 어렵다.

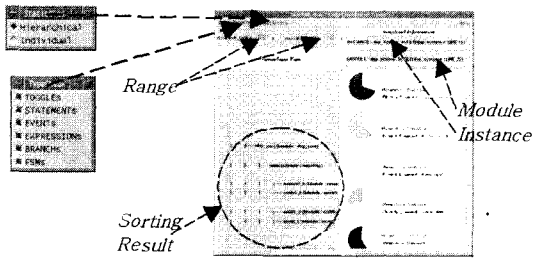
Ⅳ. Code Coverage Analysis

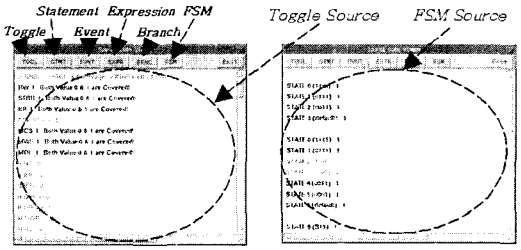
코드의 적용 범위 분석(Code Coverage Analysis)은 설계된 회로에서 검증의 충실도를 판단할 수 있는 중요 지표(Index)로써 궁극적으로 설계/검증의 질(Quality of Design & Verification)을 높일 수 있는 흐름을 제공한다. <그림 5>는 본 논문에서 소개하는 설계 환경에 통합되어 있는 코드의 적용 범위 분석 흐름이다.

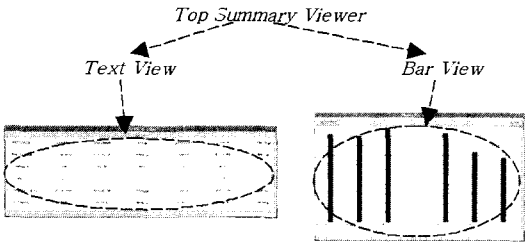
본 설계 환경은 앞의 레지스터 전송 수준의 전력 소모 예측 흐름에서 언급한 PI(Procedural Interface)의 일종인, PLI의 Verilog용 표준인 VPI를 사용하여 만들었다. 주어진 레지스터 전



<그림 5> 코드 적용 범위 분석 흐름
<**Fig. 5**> Code Coverage Analysis Flow







<그림 6> 코드 적용 범위 분석 결과 보기 툴 예제
<**Fig. 6**> Code Coverage Analysis Result Viewer

송 수준의 코드(VHDL 또는 Verilog)로부터 Toggle, Statement, Event, Expression, Branch, 그리고 FSM 정보를 추출하여, 지금 사용하고 있는 테스트 코드 또는 벡터가 적절하게 해당 항목들의 전 영역(Range)을 체크하고 있는지를 분석한다. 본 환경은 단순히 적용 범위 분석만을 수행하는 것이 아니라, 분석된 결과를 바탕으로 적은 수의 벡터를 사용하여 짧은 시간 동안에 높은 적용 범위 수준(Coverage)를 얻을 수 있도록 테스트 벡터(Test Vector: Test Suite)의 적용 순서를 조정하여 주는 테스트 벡터 최적화(Test Suite Optimization) 기능도 함께 지원한다. 뿐만 아니라 분석 결과를 쉽게 이해할 수 있도록 Tcl(Tool command language)/Tk(Toolkit)를 이용하여 작성된 GUI(Graphical User Interface) 환경의 결과 보기(Result Viewer) 툴을 함께 지원한다. 〈그림 6〉은 GUI 환경의 결과 보기 툴을 보여준다.

본 환경을 이용하여 기존의 동작 검증 위주의 시뮬레이션을 행함과 동시에 코드의 적용 범위 분석 결과도 함께 얻을 수 있고 최적의 테스트 코드 또는 벡터의 적용 순서를 정할 수 있으며, 이를 바탕으로 테스트 코드 또는 벡터의 충실도를 높여 설계/검증의 질(Quality)를 보장할 수 있다.

V. 성능 비교

본 설계 환경은 기존의 동작 검증 환경에 익숙한 설계자들이 동작 검증에 사용한 시뮬레이션에 기반한 설계 환경을 그대로 이용하여, 레지스터 전송 수준에서 전력 소모를 예측하고 코드의 적용 범위를 분석할 수 있는 흐름을 개발하여 이를 기존의 SoC Front-end 설계/검증 환경에 통합하여 효율적인 설계/검증이 가능하도록 하였다. 따라서 기존의 동작 검증만을 위한 환경, 즉 시뮬레이션만을 이용했을 때의 성능과 여기에 레지스터 전송 수준에서의 전력 소모 예측, 또는 코드의 적용 범위 분석 등의 기능을 함께 적용 하였을 때의 성능을 함께 비교하여, 기존의 전통적인 설계/검증 환경(Simulation Based)에 익숙한 설계자들이 쉽게 사용할 수 있는 환경임을 보여 준다. SUN UltraSparc(CPU : 200MHz, Main memory : 525MB) 워크스테이션을 사용하여 순수한 동작 검증에 걸리는 시뮬레이션 시간과, 본 설계 환경에 통합되어 있는 레지스터 전송 수준의 전력 소모 예측과 코드 적용 범위 분석을 함께 이용하였을 때에 걸리는 시간을 비교한 것이다.

〈표 4〉의 코드 적용 범위 분석 결과란에서 Count 계량(Measure)은 전체 레지스터 전송 수준의 코드에서 각 항목(Toggle, Statement, Event, Expression, Branch, 그리고 FSM 정보에서 State/Transition)들이 전체 동작 검증

〈표 3〉 RTL Power Estimation 성능 비교
〈**Table 3**〉 RTL Power Estimation Result

Benchmark	Verilog-XL	Verilog-XL+ RTL Power Estimation
GCD	0.1S	0.2S
AM2901	1.2S	19.3S
RS-ECC	14.3S	116.1S
LCD Driver	58.2S	290.1S
DSP	12.8S	55.7S

〈표 4〉 Code Coverage Analysis 성능 비교
〈**Table 4**〉 Code Coverage Analysis Result

Benchmark	Verilog-XL	Verilog-XL+ Code Coverage Analysis	
LCD Driver	39S	Binary	57S
		Count	207S
DSP16g1	16S	Binary	23S
		Count	58S
RS-ECC	12S	Binary	25S
		Count	68S
MP3 Decoder	129S	Binary	205S

동안에 체크되는 횟수를 모두 기록하면서 분석을 행하는 과정의 결과이고, Binary 계량은 각 체크 항목들이 한 번씩만 체크될 경우 더 이상의 체크 회수를 기록하지 않는 과정을 의미한다. 당연히 Binary 계량이 Count 계량에 비하여 분석에 걸리는 시간이 더 적음을 볼 수 있다.

위의 결과에서 알 수 있듯이 레지스터 전송 수준의 전력 소모 예측과 코드 적용 범위 분석을 수행하기 위하여, 기존의 동작 검증에 소요되는 시간을 감안하여 볼 때에 추가로 필요한 시간적 오버헤드(Overhead)는 그리 크지 않다. 따라서 기존의 동작 검증 위주의 설계 환경에 익숙한 설계자들도 새로운 툴 또는 설계 환경을 익혀야 하는 노력 없이도 본 통합 설계/검증 환경을 이용하여 쉽게 레지스터 전송 수준의 전력 소모 예측과 코드 적용 범위 분석을 행할 수 있음을 알 수 있다.

Ⅳ. 결 론

본 논문에서는 32-bit 프로세서 코어 내장형 SoC를 개발할 때 실제로 유용하게 이용되고 있는 Front-end 통합 칩 검증(Chip Validation) 환경을 소개하였다. 시뮬레이션에 기반한 전통적인 동작 검증 환경을 기반으로 하고 있기 때문에, 기존의 전통적인 설계/검증 환경(Conventional Design & Verification Environment)에 익숙한 설계자들이 추가적인 노력과 시간의 투자가 없이도 쉽게 레지스터 전송 수준에서 전력 소모 예측과 코드의 적용 범위 분석을 동시에 수행할 수 있도록 만들어져 있으며, 이 외에도 C 컴파일러(armcc) 또는 어셈블러(armasm)를 사용한 레지스터 전송 수준에서의 설계/검증은 물론, 여기에 사용된 소스 코드(Source Code : C Codes, Header Files, Libraries)들을 그대로 이용하여 Back-annotation 정보를 이용한 게이트 수준 이하의 검증에도 이용할 수 있다. 또한 칩 제조 이후의 테스트 단계에서 이용할 수 있도록 테스트 벡터를 생성하는 기능도 함께 가지고 있는 통합 설계/검증 환경이다.

본 통합 환경을 사용하여 SoC VLSI 설계의 맨 처음 단계인 레지스터 전송 수준에서의 설계/검증의 질(QoR : Quality of Result)을 한 단계 끌어 올릴 수 있으며, 궁극적으로 SoC VLSI 설계를 짧은 시간 내에 오류 없이 완료할 수 있다.

참 고 문 헌

〔1〕 PowerRater(Hynix In-house RTL Power Estimat)-김기선, 안태균 Hynix Semiconductor Inc. 2000

〔2〕 CoverRater(Hynix In-house RTL & Gate-level Code Coverage Analyzer)-김기선, 안태균 Hynix Semiconductor Inc. 2000

〔3〕 P. E. Landman and J. M. Rabaey, "Architectural Power Analysis : The Dual Bit Type Method," IEEE Trans. VLSI Systems, vol. 3, pp.173-187, June 1995.

〔4〕 A. Raghunathan, S. Dey, and N. K. Jha, "Register-Transfer Level Estimation Techniques for Switching Activity and Power Consumption," International Conference on Computer-Aided Design, pp.158-165, Nov. 1996.

〔5〕 C.-T. Hsieh, Q. Wu, C.-S. Ding, and M. Pedram, "Statistical Sampling and Regression Analysis for RT-Level Power Evaluation," International Conference on Computer-Aided Design, pp.583-588, 1996.

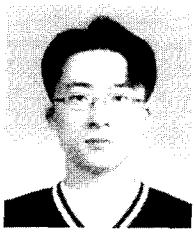
〔6〕 Q. Wu, Q. Qiu, M. Pedram, and C.-S. Ding, "Cycle-Accurate Macro-Models for RT-Level Power Analysis," IEEE Trans. VLSI Systems, Dec. 1998.

〔7〕 Z. Chen and K. Roy, "A Power

Macromodeling Technique Based on Power Sensitivity," Design Automation Conference, pp. 678-683, 1998.
〔8〕 J.-Y. Lin, W.-Z. Shen, and J.-Y. Jou, "A Power Modeling and Characterization Method for Macrocells Using Structure Information," International Conference on Computer-Aided Design, pp.502-506, 1997.
〔9〕 T. Ahn, K. Choi, K.-H. Kim, and S.-K. Hong, "A New Cost Model for High-Level Power Optimization and its Application," International Symposium on Circuits and Systems, pp.II573-II 576, 2000.
〔10〕 PCALC(Hynix In-house Gate-level Power Estimation Tool)-한상열 Hynix Semiconductor Inc. 2000

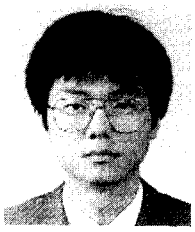
* ARM은 Advanced RISC Machine社의 등록 상표입니다.

저 자 소 개

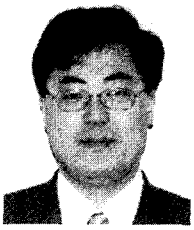


김 기 선
1997년 2월 한양대학교 공과대학 전자공학과 졸업(학사), 1999년 2월 서울대학교 대학원 전기공학부 졸업(석사), 1999년 3월~1999년 11월 : 엘지반도체 램버스 개발팀, 1999년 12월~2000년 12월 : 현대전자 CAD팀, 2001년 1월~현재 : Hynix 반도체 MCU-SOC팀, 〈주관심 분야 : SoC, Desgn Automation〉.

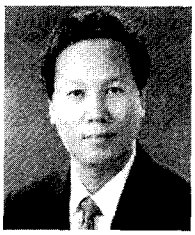
김 성 식
1988년 2월 경북대학교 전자공학과 학사, 1999년 8월 충북대학교 정보통신공학과 석사, 2003년 8월 충북대학교 정보통신공학과 박사, 1988년 1월~현재 : Hynix 반도체 MCU설계



이 희 연
1984년 2월 서강대학교 전자공학과 (학사), 1986년 2월 서강대학교 대학원 전자공학과 (석사), 1986년 1월~현재 : Hynix 반도체 MCU-SOC팀



김 기 현
1982년 2월 한양대학교 공과대학 전자공학과 졸업, 1986년 2월 한양대학교 대학원 전자공학과 졸업(석사), 1996년 2월 한양대학교 대학원 전자공학과 졸업(박사), 1986년 2월~1994년 5월 : 전자통신연구원(ETRI) 연구원, 1996년 2월~현재 : Hynix 반도체 수석연구원,



채 재 호
1978년 2월 연세대학교 전자공학과 학사, 1981년 2월~현재 : Hynix 반도체 MCU 담당