시스템 반도체 설계 교육에 “내 칩 MPW 서비스”와 오픈-소스 EDA 도구의 활용

국일호 [1], 박건식 [2]
경희대학교 [1], 전자통신연구원 [2]
e-mail : goodkook@khu.ac.kr[1], kunsik@etri.re.kr[2]

Using Open-Source EDA Tools and “My Chip MPW Service” for SoC Design Education

Kook,ilho[1] and Park, Kunsik[2]
Kyunghee University[1], ETRI[2]

Abstract

The difficulties in SoC design education are that expensive design tools and the secrecy of PDK make it impossible to provide experimental training course. Recently, the open-source movement has spread to EDA field, increasing the usability of design tools. In addition, "My Chip MPW service" has been provided with the purpose of fostering system semiconductor design engineer. The service fabricate student’s design and deliver packaged IC at no charge. These have resolved two major obstacles to semiconductor design education. This paper proposes a standard cell-based design kit targeting the NSPL050 process using open-source EDA tools and a method to utilize it in semiconductor design education. The design flow presented in this paper covers the entire process from system-level SystemC/C++ test bench, Verilog RTL design and verification, synthesis, and automatic placement and routing, finally fab-ready GDS generation. The layout generated through the design flow was cross-validated with commercial DRC tools. For the education and validation purpose the example designs are also included in the design kits.

I. 서론

증가하는 시스템 반도체의 설계 규모와 복잡도에 대응하기 위하여 높은 추상화 수준에서 시작하는 설계 방법론이 적용되어 발전하고 있다[2]. 대학의 교육 역시 이에 맞춰 하드웨어 언어 기반의 설계 방법론을 추구하며 시뮬레이션과 FPGA 를 활용한 실습이 주를 이룬다. 시스템 반도체 설계 과목은 전자회로와 프로그래밍 언어 그리고 레이아웃을 다루는 VLSI 설계를 먼저 이수할 것을 요구 한다[3]. 하지만 선수 과목들과 시스템 수준 설계 방법론 사이에 추상화 수준의 격차가 크고 학습 영역의 폭이 넓어 이를 체계적으로 학습할 수 있는 실습 과정을 마련하지 못하고 있다. 특히 ASIC 설계 실습을 어렵게 하는 가장 큰 애로 사항으로 EDA 도구의 사용에 있어서 고가의 라이센스 비용, 지나치게 전문적인 사용법, 목표 공정 자료(PDK)의 폐쇄성 등이 꼽힌다. 수년 간 IDEC 을 통해 교육용 라이센스를 제공하고 있지만 연구 목적에 사용하기도 부족하다. 공개된 공정 자료가 있더라도 이를 상용 도구에 이식하려면 상당한 비용을 지불해야 한다.

본 논문은 오픈-소스 EDA 도구를 시스템 반도체 설계 교육에 활용하는 방법을 제안한다. 표준-셀과 함께 “내 칩 MPW 서비스[1]”의 공정(이하 NSPL050)을 오픈-소스 EDA 도구들에 이식하고 통합한 설계 환경(이하 디자인 킷)을 마련하였다. 오픈-소스 설계 도구들을 사용함으로써 개방된 공정 정보의 활용이 수월할 뿐만 아니라 무엇보다도 도구 사용 비용을 지불하지 않고 “ 언제 어디 서나” 시스템 반도체 설계를 학습 할 수 있다. 제안하는 디자인 킷에는 표준-셀 라이브러리를 포함하여 공정 전용 사인-오프 유틸리티도 포함한다. 아울러 알고리즘 분석, 구조 탐색 및 “내 칩 MPW” 요건에 맞도록 분할, 시스템 수준 테스트벤치, 베릴로그 시뮬레이션, 합성과 자동 배치배선, 사인-오프 까지 수행해 볼 수 있는 예제도 제시하였다. 오픈-소스 EDA 도구들을 모아 툴-체인을 완성하는 과정에서 파이썬(python)과 티클(Tcl), Makefile, Bash-Shell 등 스크립트 들이 활용되었다. 디자인 플로우의 검증 기법은 베릴로그와 SystemC/C++ 코-시뮬레이션, FPGA 활용 코-에뮬레이션, 출고된 칩의 테스트까지 전 과정을 아우른다. 본 디자인 킷을 활용하여 생성된 레이아웃 GDS 는 상용 도구[25]와 디자인 룰을 교차 검증 하였다.

II. 본론

시스템 반도체 설계 교육과정의 과목간 불연속성의 원인을 살펴보고 이를 극복할 방안으로 구축한 오픈-소스 EDA 도구를 활용한 설계 환경을 제안한다.

II-1. 반도체 설계 교육에서 과목 간 연속성 단절

반도체 부품이 만들어지기까지 여러 단계의 추상화 수준 전환을 거친다. 대학의 교육 과정은 표면적으로 이에 소요되는 내용을 모두 다루고 있다. 하지만 과목 간의 연속성이 미흡하다. 통합된 실습 과정이 없다는 점이 분절된 과목들 사이에 간격이 넓어진 원인으로 한몫한다. 반도체 설계 실습을 어렵게 만드는 가장 큰 장애 요인으로 고가의 설계도구(소프트웨어)와 공정비용을 감당하기 어렵다는 점을 꼽을 수 있다. 상용 설계도구의 지나치게 복잡한 사용법과 비밀주의 그리고 상업주의 또한 무시할 수 없다. 이로 인하여 반도체 부품의 사용이 이미 일반화 되었음에도 전용 칩(ASIC) 설계의 교육 과정은 이론적 수준에 머물러 있다.

II-2. "내 칩 MPW 서비스" 개황

"내 칩(My Chip) MPW 서비스"는 반도체 설계 인력 양성을 목적으로 2023 년에 시작되었다. 전자통신연구소(ETRI)의 반도체 실험실(NSPL)[37]을 비롯하여 대학 연구소(ISRC[35], DGIST[36])에 설치된 실리콘 제조 공정을 활용하여 학생들의 설계를 반도체 부품으로 무료 제작해 주는 획기적인 공공 서비스다. 2024 년 현재 2 차에 걸친 공정을 완료하여 패키지를 배포하였고 2, 3 회 차 공정이 진행 중이다. 이 서비스를 통해 제공되는 공정의 PDK 는 반도체 제조 도면 그리기 규칙(디자인 룰)과 물성 인자(SPICE 파라메터) 외에 아날로그 PCELL 그리고 GPIO 셀을 제공한다.

II-3. 오픈-소스 EDA 도구의 동향

이미 80 년대부터 공개된 반도체 설계 도구로 레이아웃 도구 Magic[24], 회로 시뮬레이터 SPICE[17]가 있었다. 최근 소프트웨어 분야의 오픈-소스 운동이 큰 성과를 거두었고 반도체 설계 자동화(EDA) 도구 분야에도 확산되고 있다[6][7]. 하드웨어 언어 기반의 디지털 회로 설계가 정착되면서 하드웨어 언어 시뮬레이터의 사용이 필수가 된 지금 오픈-소스 시뮬레이터 iVerilog[13], 베릴로그 언어 변환기 Verilator[19]등은 HDL 언어 표준을 대부분 수용하는 수준에 이르렀고, 시스템 모델링 및 시뮬레이션 용 C++ 라이브러리 인 SystemC 는 표준으로 정착되었다[9]. 표준 셀 라이브러리 규격과 도구 사이의 파일 규격 LEF/DEF[30] 역시 표준화 하면서 상용 도구를 보조하던 실험실 내 도구들이 공개되어 독립적으로 발전하였다. 레이아웃 외에 RTL 합성과 논리 최적화[21], 자동 배치[22] 및 배선[23] 도구 들이 오픈-소스화 하였다. 이들 오픈-소스 도구들은 상당 부분 공공 지원을 받은 학술 연구의 소산이기도 하다. SkyWater[31], GF[32] 등 최첨단은 아니지만 여전히 유용한 상용 공정들의 PDK 가 공개되고 오픈-소스 EDA 에 이식되자 완성도는 더욱 향상되었다. 설계 과정의 각 추상화 단계마다 개별적으로 사용되던 요소 도구들이 통합된 설계 환경으로 발전하게 되었다[26][27][28]. 공개된 공정을 활용하여 교육 및 MPW 를 대행하는 eFabless 사[33]가 운용 중이며 교육 뿐만 아니라 '메이커' 활동에도 적은 비용으로 내 칩을 만들고 있다[34]. 교육과 혁신적 아이디어의 구현에 공정의 첨단성은 크게 중요치 않다.

II-4. 오픈-소스 EDA 도구 활용 "디자인 킷"

"내 칩 MPW 서비스"는 기본적으로 "풀-커스텀" 설계의 레이아웃 GDS 를 접수 받아 칩을 제작해 준다. 일반적으로 MPW 서비스는 공정을 저비용으로 제공 할 뿐이며 설계 방법론은 설계자들의 선택이다. 하드웨어 언어에 기반한 설계 방법을 지원하기 위해 필요한 표준 셀은 모두 검증되고 보장된 IP 라는 점을 감안하면 디자인 킷을 준비하기 위해 엄청난 비용이 소요된다. 뿐만 아니라 공공 서비스로서 특정 상용 설계 도구의 사용을 강요할 수는 없다. "내칩 MPW 서비스"의 NSPL050(2 중 폴리실리콘 3 중 금속 표준 CMOS)공정을 목표로 하드웨어 언어로 작성된 설계를 ASIC 으로 구현할 때 쓰일 디자인 킷을 제작하였다. 이 서비스를 통해 칩 제작은 물론 패키지까지 무료로 제공받길 원할 경우 입출력 패드를 제외한 순수 설계 면적은 1000x1000um, 그리고 28 개의 입출력 핀 제한을 따라야 한다. HDL 합성 디지털 회로를 고려하는 경우 부족 할 수 있으나 제약을 극복하는 과정에서 새로운 아이디어와 혁신이 나온다. 알고리즘 분석과 회로 분할 또한 연구 활동이라고 할 것이다.

본 논문에서 제안하는 "디자인 킷"은 고가의 설계 도구들을 배제하고 오픈-소스 도구들을 활용하였다. NSPL050 의 PDK 에 따라 작성된 표준 셀들은 SPICE 회로 시뮬레이션[15]을 수행 하였고 추상화 수준마다 요구하는 형식으로 기술되었다. 개별적인 오픈-소스 EDA 도구들 사이의 연계 용 스크립트들과 공정 특성에 맞는 사인-오프 툴 들로 구성되었다. 그림 1 은 본 논문에서 제안 하는 오픈-소스 설계 자동화 도구들을 활용한 표준 셀 기반 디지털 반도체 설계의 플로우를 보여준다. 높은 추상화 수준에서 칩 제작에 필요한 도면의 생성까지 전 단계에 이르는 과정에 모두 오픈 소스 도구들을 활용하였다. 공정을 마친 칩을 패키지로 제공 하며 이를 시험하는 과정까지 포함한다. 본 논문에서 제안하는 설계 환경에서 사용된 오픈-소스 도구들의 목록과 용도는 표 1 과 같다. RTL 에서 시작한 하드웨어 설계가 반도체 제작 가능한 레이아웃으로 변환되기까지 추상화 낮춤 과정을 거치게 되며 그때마다 사용되는 도구도 다르다. 자동화 도구를 활용하는 설계의 과정은 반드시 검증 절차가 수반되어야 한다. 서로 다른 수준에서 자동화 도구가 낳은 결과가 일치하는지 확인 해주는 테스트 벤치는 검증의 기준 틀이다. 테스트 벤치는 RTL 보다 높은 시스템 수준에서 작성토록 하였다. 언-타임드 및 타임드 시스템 모델링을 위해 C++언어의 크래스 라이브러리 SystemC[9]를 활용 한다. 테스트벤치의 작성에 최상위 추상화 수준의 C++언어를 이용함으로써 이미 쌓여있는 수많은 수학 라이브러리[10], 데이터 시각화 도구들[12]은 물론 대화형 시뮬레이션이 가능한 다양한 자원들[11]을 하드웨어 설계의 검증에 활용 할 수 있다. SystemC 로 작성된 테스트 벤치는 RTL 설계가 추상화 단계를 낮출 때마다 검증을 위해 재사용 되며 최종적으로 공정을 마친 칩의 테스트 용도로도 쓰인다. 하드웨어의 추상화 수준이 전환되는 단계마다 사용될 도구들을 통합한 환경은 QFlow[26]다. 본 논문에서 제안하는 오픈-소스 EDA 도구의 설계 환경은 "내칩 MPW 서비스"의

공정을 QFlow 에 이식한 것이다.

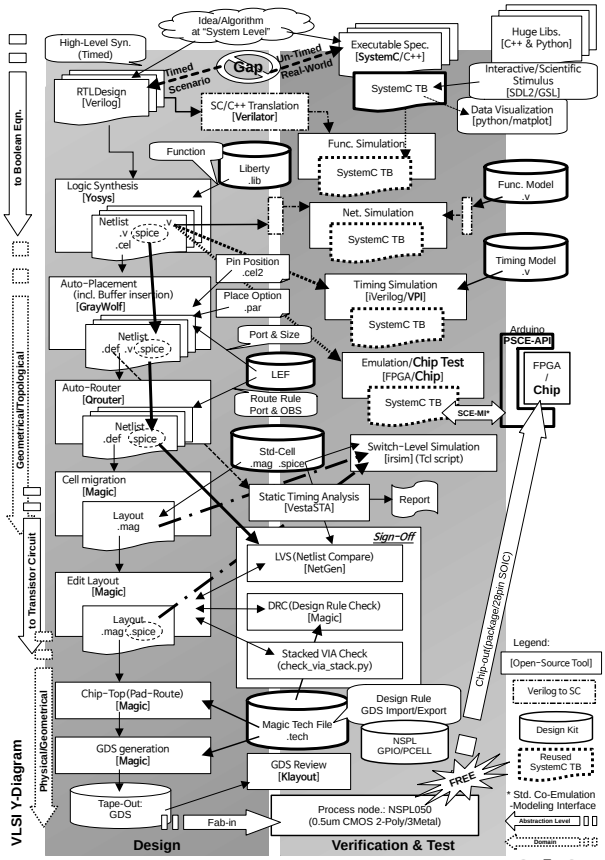

그림 1. 본 논문에서 제안 하는 오픈-소스 EDA 도구들을 활용한 표준-셀 디지털 반도체 설계 플로우

표 1. 디자인 킷에 활용된 오픈 소스 도구 및 라이브러리 목록

도구 명	용도
SystemC[9]	시스템 수준 모델링 C++ 크래스 라이브러리
GSL[10]	C/C++ 과학 라이브러리(GNU Scientific Library) 시뮬레이션 데이터 생성 및 분석
SDL2[11]	멀티미디어 C/C++ 라이브러리(Simple Directmedia Layer). 대화형 테스트 벤치
python[12]	시뮬레이션 데이터 분석 및 시각화
iVerilog[13]	베릴로그 시뮬레이터
irsim[16]	스위치 레벨 시뮬레이터
ngSpice[15]	SPICE 회로 시뮬레이터
XSchem[18]	회로도 작성
Verilator[19]	베릴로그-C++ 변환기
Yosys[20]	RTL 베릴로그 합성기
GrayWolf[22]	자동 배치 배선기. 버퍼 삽입 기능 포함
Qrouter[23]	자동 배선기
Magic[24]	레이아웃 편집, DRC, GDS 생성
NetGen[25]	네트리스트 비교 방식 LVS
QFlow[26]	설계 플로우 관리 도구

표 2. 디자인 킷의 설계 자동화 용 표준-셀 구성과 파일 형식

파일 명	형식
	관련 도구(용도)
etri050_stdcells.lib	리버티(liberty)
	RTL 합성(function)
etri050_stdcells.lef	LEF
	배치(PIN, SIZE)
	배선(PIN, OBS)
etri050_stdcells.sp	SPICE
	LVS
SCN3ME_SUBM. 30.ETRI.tech	Magic Tech.
	DRC, 회로 추출 GDS 들여오기/내보내기
etri05_stdcells.v	베릴로그
	기능 시뮬레이션 모형
etri05_stdcells_func.v	베릴로그
	타이밍 시뮬레이션 모형
*.mag	표준 셀 레이아웃

표 3. 디자인 킷의 표준-셀 종류

표준-셀	기능(리버티 형식[29])
AND2X1, AND2X2	(A B)
AOI21X1	(!((A B)+C))
AOI22X1	(!((A B)+(C D)))
BUFX2,BUFX4	A
CLKBUF1,CLKBUF2	A
DFFNEGX1	ff() { …… }
DFFPOSX1	ff() { …… }

DFFSR	ff (P0002,P0003) {
	next_state : "D";
	clocked_on : "CLK";
	clear : "(!R)";
	preset : "(!S)";
	clear_preset_var1 : L;
	}
	
	“P0002”

INVX1,INVX2,INVX4	(!A)
MUX2X1	(!((S A) + (!S B)))
NAND2X1,NAND2X1	(!(A B)),(!((A B) C))
NOR2X1, NOR2X1	(!(A+B)), (!((A+B)+C))
OAI21X1,OAI22X1	(!((A+B) C)),(!((A+B) (C+D)))
OR2X1, OR2X2	(A+B)

표준-셀은 합성의 기초 기능 요소이자 자동 배치배선의 기본 부품이다. 자동화 도구는 금속 층을 사용하여 배치와

배선을 할 뿐이며 표준-셀의 물리적 특성에 따라 칩의 성능이 결정된다. 본 "디자인 킷"은 합성기에서 요구하는 가장 기본적인 논리 기능을 가진 표준 셀을 제공하였다. VLSI 설계 과목에서 실습하는 레이아웃 그리기 연습을 넘어 저마다 고유의 표준 셀 라이브러리를 꾸며 보는 것도 의미 있을 것이다. 논리 식을 트랜지스터 회로로 표현 하고 이를 자동화 도구의 규격에 맞는 레이아웃을 그려보는 것이다. 표 3 는 본 디자인 킷에서 제공하는 기본 표준-셀의 목록과 리버티[39] 양식의 기능 표현을 나타내었다.

설계 자동화를 이루기 위해 표준-셀은 기본적인 논리 기능을 여러 추상화 수준에서 표현 되어야 한다. 추상화 수준이 전환될 때마다 사용되는 자동화 도구에 적용하려면 이에 요구하는 규격을 만족해야 하기 때문이다. 합성기용 리버티 [39], 자동 배치와 배선 도구의 LEF[40], 시뮬레이션 모델 등이 표준 셀의 레이아웃과 함께 준비되어야 한다. 합성과 자동 배치와 배선 그리고 시뮬레이션을 위한 표준-셀의 구성 파일 형식은 표 2 와 같다.

표 4. 디자인 킷을 적용한 설계 예제의 자동화 도구 결과 비교

설계	FIR-PE		ALU8	
합성/ 표준-셀	AND2X2 AOI21X1 AOI22X1 BUFX2 **DFFPOSX1** INVX1 NAND2X1 NAND3X1 NOR2X1 OAI21X1 OAI22X1 OR2X2	30 74 21 9 75 126 145 94 85 175 6 10	AND2X2 AOI21X1 AOI22X1 BUFX2 DFFPOSX1 **DFFSR** INVX1 MUX2X1 NAND2X1 NAND3X1 NOR2X1 NOR3X1 OAI21X1 OAI22X1 OR2X2	40 95 37 9 24 3 139 3 171 145 63 4 219 7 17
배치 배선	Density=1.0 Size: 736 x 697um		Density=0.95 Size: 786 x 733um	
사인- 오프	LVS: Pin/Net Match DRC: None Stacked VIA: 1		LVS: Pin/ Net Match DRC: None Stacked VIA: 3	

본 논문에서 제안한 디자인 킷의 유효성을 확인하기 위해 두 가지 예제를 제시한다. 이 예제들은 베릴로그 소스 코드는 물론 설계 도구들에 주어질 각종 환경 파일들이 포함되어있다. 여러 단계에서 자동화 도구의 실행에 복잡한 명령-줄 인수가 주어지는데 이는 해당 도구의 사용법이기도 하다. 예제에 Makefile 을 두어 학습에 활용 할 수 있도록 하였다.

첫 번째 예제는 8-탭 FIR 필터다. 8 비트 디지털 샘플 입력을 받아 저역통과 필터링을 수행 한다. MPW 의 허용 면적에 제한이 있으므로 필터 전체를 넣을 수 없다. C++ 언어의 for 반복문으로 기술된 필터 알고리즘을 파이프 라인 병렬처리 구조로 도출하고 SystemC 로 작성한 테스트 벤치에서 언-타임드(un-timed) 시뮬레이션으로 검증 하였다. 파이프라인을 구성하는 처리요소(PE, processing Element)는 8 비트 곱셈기와 16 비트 누산기로 구성되었다. 베릴로그 RTL 로 기술된 PE 는 SystemC 테스트 벤치와 코-시뮬레이션으로 검증 하였다. 오픈-소스 베릴로그 시뮬레이터 iVerilog 가 VPI 를 지원 하므로 합성 후 타이밍 시뮬레이션에도 SystemC 테스트 벤치를 재사용 할 수 있다. 두 번째 예제는 8 비트 ALU 로 논리 연산 및 가감산기와 곱셈기를 갖추고 있다. 범용 연산기는 데이터 패스 구조를 가지게 되어 많은 입출력 핀이 필요하다. "내 칩 MPW"의 입출력 핀 수 제한을 해소하기 위한 방안으로 외부 인터페이스용 핸드쉐이크 기구를 유한 상태 머신(FSM, Finite State Machine)으로 구현 하였다. 앞 예제의 파이프라인 PE 는 단순 플립-플롭 셀 DFFPOSX1 만 사용해도 되었지만 FSM 의 특성상 리셋을 가진 DFFSR 셀이 사용 된다. 표 4 는 두 예제를 디자인 킷에 적용한 과정에서 얻어진 중간 결과를 보여준다. RTL 합성으로 사용된 표준 셀의 내역을 볼 수 있다. 표준 셀의 배치 밀도를 조절해가며 자동 배선의 성공을 얻게 되는데 전역 신호로서 리셋이 들어간 경우 배치 밀도가 다소 낮아 자동 배선에 불리함을 알 수 있다.

자동 배치와 배선 도구 역시 오류(버그)에서 자유롭지 못하는 소프트웨어다. 자동화 도구에서 생산된 결과물은 다른 도구를 사용하여 교차 검증 되어야 한다. 본 디자인 킷은 자동 배선 후 생성된 레이아웃을 검사하는 사인-오프 절차를 두고 있다. 두 예제의 레이아웃을 DRC, LVS 그리고 적층 비아 검사를 수행한 결과 역시 표 4 에서 볼 수 있다. LVS 는 합성 후 이어진 자동 배치에서 버퍼를 삽입한 네트리스트와 레이아웃에서 추출한 SPICE 형식 네트리스트를 비교한 것이다. NSPL050 공정은 적층 비아(stacked VIA)를 허용하지 않는다. 자동 배선 도구의 옵션으로 적층 비아를 금지 시켜 놓았으나 이를 완벽히 피하지 못한다. 디자인 킷에 이를 검사하기 위한 파이썬 프로그램을 마련해 두었다. 검사 결과 두 예제에서 모두 적층 비아가 발견되었다. 이 검사 프로그램은 적층 비아를 검출 하고 레이아웃 상에 이를 표시해 줄 수 있을 뿐이므로 수동 편집으로 해소 되어야 한다.

공정의 정밀도에 따라 규정된 디자인 룰은 설계의 레이아웃에서 반드시 준수해야 할 사항이다. 적층 비아의 금지, 배선 금속과 비아의 폭과 이격, 다른 전위를 갖는 디퓨젼 이격은 자동 배치와 배선 도구에서 유발될 수 있는 오류다. 예제의 두 설계에 대하여 디자인 킷을 적용하여 얻은 레이아웃 GDS 를 상용의 도구[28]로 교차 DRC 검증을 수행 하여 동일한 로그를 얻었다.

III. 결론 및 향후 연구 방향

본 논문은 시스템 반도체 설계 교육에서 과목 간 학습 불연속성의 원인을 살펴보고 이를 극복할 방안으로 오픈-소스 EDA 도구를 활용한 설계 환경을 제안하였다. 반도체 설계 도구의 사용과 공정 비용은 시스템 반도체 설계를 실습에 가장 큰 장애 였으나 무료 "내 칩 MPW 서비스"와 오픈-소스 EDA 도구는 이를 일거에 해결 할 수 있다. NSPL050 공정은 아날로그 회로를 포함된 혼성 신호 설계도 가능 하도록 PDK 에 BJT, DIODE, PIPCAP, P2RES 등의 아날로그 PCELL 들을 제공한다. 앞으로 합성 방식 디지털 회로 뿐만 아니라 아날로그 회로가 포함된 혼성 신호 반도체 디자인 킷을 마련 할 것이다.

본 논문의 “표준-셀 디자인 킷”은 경희대학교 반도체 전공 트랙 사업단의 연구 성과로서 깃-허브 저장소를 통해 공개하였다[38]. ETRI/반도체 실험실의 협조를 받아 “내 칩 MPW 서비스”를 통해 실리콘 검증을 준비하고 있다. 향후 시스템 반도체 설계 인력 부족이 심각한 지금 시스템 반도체 설계 인력 양성에 기여 할 것으로 기대한다.

참고문헌

[1] 국가 나노 인프라 협의체 내칩(My chip)제작 서비스, http://mpw.kion.or.kr/
[2] Towards Automatic High-Level Code Deployment on Reconfigurable Platforms: A Survey of High-Level Synthesis Tools and Toolchains, https://ieeexplore.ieee.org/document/9195872
[3] 전자공학 이수 체계도, https://ee.khu.ac.kr/ee/user/contents/view.do?menuNo=1900018
[4] A Mixed Open-Source and Proprietary EDA Commond for Education and Prototyping, https://ieeexplore.ieee.org/document/10069012
[5] Bringing Academic EDA to Real-World Usability, https://ieeexplore.ieee.org/document/9256694
[6] The OpenROAD: Toward a Self-Driving, Open-Source Digital Layout Implementation Tool Chain, https://vlsicad.ucsd.edu/Publications/Conferences/370/c370.pdf
[7] OpenLANE: The Open-Source Digital ASIC Implementation Flow, https://woset-workshop.github.io/PDFs/2020/a21.pdf
[8] Standard Cell Library Design and Optimization Methodology for ASAP7 PDK, https://ieeexplore.ieee.org/document/8203890
[9] SystemC Language Reference Manual, https://systemc.org/resources/standards/
[10] GSL - GNU Scientific Library, https://www.gnu.org/software/gsl/
[11] SDL: Simple DirectMedia Layer, https://www.libsdl.org/
[12] Python Programming And Numerical Methods: A Guide For Engineers And Scientists, https://pythonnumericalmethods.studentorg.berkeley.edu/notebooks/Index.html
[13] The ICARUS Verilog Compilation System, https://github.com/steveicarus/iverilog
[14] IRSIM switch-level simulator for digital circuits, https://github.com/RTimothyEdwards/irsim
[15] ngspice - open source spice simulator, https://ngspice.sourceforge.io/
[16] IRSIM switch-level simulator for digital circuits, https://github.com/RTimothyEdwards/irsim
[17] ngspice - open source spice simulator, https://ngspice.sourceforge.io/
[18] XSCHEM : schematic capture and netlisting EDA tool, https://xschem.sourceforge.io/stefan/index.html
[19] VERILATOR, https://www.veripool.org/verilator/
[20] Yosys Open Synthesis Suite, https://yosyshq.net/yosys/
[21] ABC: A System for Sequential Synthesis and Verification, https://people.eecs.berkeley.edu/~alanmi/abc/abc.htm
[22] GrayWolf, https://github.com/rubund/graywolf
[23] Qrouter, http://opencircuitdesign.com/qrouter/
[24] Magic VLSI Layout Tool, http://opencircuitdesign.com/magic/
[25] Netgen netlist comparison (LVS) and format manipulation, http://opencircuitdesign.com/netgen/
[26] Qflow: An Open-Source Digital Synthesis Flow, http://opencircuitdesign.com/qflow/index.html
[27] OpenROAD Project, https://theopenroadproject.org/
[28] OpenLane, https://github.com/efabless/openlane2
[28] Calibre DRC, https://eda.sw.siemens.com/en-US/ic/calibre-design/physical-verification/nmdrc/
[29] Liberty Reference Manual (Version 2007.03), https://people.eecs.berkeley.edu/~alanmi/publications/other/liberty07_03.pdf
[30] LEF/DEF Language Reference, https://www.ispd.cc/contests/18/lefdefref.pdf
[31] SkyWater130 PDK, https://www.skywatertechnology.com/sky130-open-source-pdk/
[32] Global Foundaries, https://gf.com/
[33] eFabless/chipIgnite, https://efabless.com/
[34] Tiny Tapeout, https://tinytapeout.com/
[35] 서울대학교 반도체공동연구소(ISRC), https://isrc.snu.ac.kr/
[36] 대구경북과학기술원(DGIST) 나노기술연구부, https://www.dgist.ac.kr/nanotech/
[37] 전자통신연구원 반도체 실험실(NSPL), https://www.etri.re.kr/semilab/kor.do
[38] 경희대학교 “표준-셀 디자인 킷”, https://github.com/GoodKook/ETRI-0.5um-CMOS-MPW-Std-Cell-DK.git