SHA-3 후보들의 H/W 구현에 대한 전력 소모량 추정*

이동건, 추상호, 김슬아, 김호원
부산대학교 컴퓨터공학과
e-mail : guneez@pusan.ac.kr

A Report on Power Estimation of SHA-3 Candidates H/W Implementation

Donggeon Lee, Sangho Chu, Seul-A Kim, Howon Kim
Dept of Computer Science & Engineering, Pusan National University

요 약

2005년 중국의 Wang 등이 SHA-1의 충돌쌍 공격에 대한 취약성을 발표한 이후 미국의 NIST(National Institute of Standards and Technologies)에서는 새로운 표준 해쉬 함수에 대한 필요성을 제기하였으며, SHA-3로 사용될 새로운 해쉬 함수를 공모하게 되었다. 전세계에서 64개의 후보들이 제안되었으며, 1라운드가 끝난 2010년 현재 14개의 후보들에 대한 2 라운드 심사가 진행중이다. 본 논문에서는 ASIC(Application Specified Intergrated Circuit) 설계 과정에서 설계 대상의 전력 소모량을 추정하는 과정을 소개하고, 이를 이용해 SHA-3 후보들의 H/W 구현들에 대해서 전력 소모량을 추정하여 결과를 제시한다.

1. 서론

지난 2007년 미국의 NIST는 새로운 해쉬 알고리즘인 SHA-3 개발을 위한 프로젝트를 시작하였다[1]. SHA-1 알고리즘은 안전한 해쉬 알고리즘이라 믿어졌지만, WANG 등[2][3]이 충돌쌍 공격을 발표하면서 SHA-1 알고리즘은 사용되지 않도록 권고되었다. SHA-2에 대한 공격은 아직 발표되지 않았지만, SHA-1과 설계 방법이 비슷하여 차후 공격이 발표될 경우 대체할 알고리즘이 필요하게 되었다. NIST는 새로운 SHA-3를 자체적으로 개발하지 않고, 후보 알고리즘을 공모한 후 가장 안전하고 효율적인 알고리즘을 선정하기로 하였다. 지난 2008년 전 세계에서 64개의 알고리즘이 응모되었으며, 이중 51개의 후보가 최소한의 제출 요건을 만족하였다. 이중 10개의 알고리즘은 제안자가 스스로 철회하였으며, 총 41개의 후보가 1 라운드 심사를 받았고, BLAKE, BLUE MIDNIGHT WISH(BMW), CubeHash, ECHO, Fugue, Grøstl, Hamsi, JH, Keccak, Luffa, Shabal, SHAvite-3, SIMD, Skein 등 14개의 후보가 1 라운드를 통과하여 2 라운드 경합을 벌이게 되었다. 2010년 하반기에는 이 중 5개의 후보로 그 대상이 더욱 좁아질 예정이며, 최종 SHA-3 후보는 2012년에 발표될 예정이다.

NIST 에서는 후보를 선정함에 있어 암호학 적인 안전성(security), 비용과 성능(cost and performance), 그리고 알고리즘과 구현 특성(algorithm and implementation characteristics)을 평가 기준으로 삼았다. 각 후보들의 하드웨어 성능에 주목한 곳이 있었는데, 일본의 AIST(Advanced Industrial Science and Technology) 내에 있는 RCIS(Research Center for Information Security)에서는 SASEBO(Side-channel Attack Standard Evaluation Board 플랫폼을 이용하여 14개 후보에 대한 하드웨어 성능을 분석하였다[4].

컴퓨팅 환경이 유비쿼터스화 되어가고, 모바일 컴퓨팅 환경이 중요시 됨에 따라 저전력 설계에 대한 중요성이 날로 부각되고 있다. 본 논문에서는 각 후보들의 하드웨어 설계를 ASIC 공정에 적용하여 Gate-Level Simulation을 통해 논리회로의 스위칭 활동을 기록, 분석하여 각 후보의 전력 소모 특성을 살펴보고자 한다.

본 논문의 나머지는 다음과 같이 구성되어 있다. 2장에서는 전력 소모량을 추정하기 위한 과정을 소개하며, 3장에서는 실험에 사용된 파라메터를 소개 하며, 4장에서는 각 후보에 대한 전력 소모량 추정 결과를 제시하며, 5장에서 글을 맺는다.

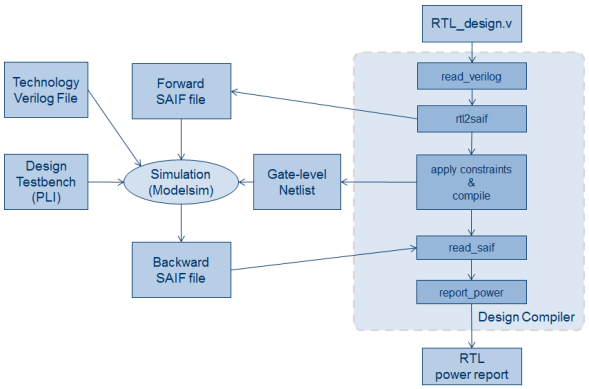
2. 전력 소모량 추정 과정

전력 소모량을 추정하기 위해 Synopsys 사의 Design Compiler와 MentorGraphics 사의 Modelsim이라는 툴을 사용하였다. Design Compiler는 RTL(Register Transfer Level)로 작성된 코드를 Gate-level Netlist로 변환시켜 주

* 이 논문은 2010년 교육과학기술부로부터 지원받아 수행된 연구임 (지역거점연구단육성사업/차세대물류IT기술연구사업단)

는 ASIC 설계 툴이다. Design Compiler는 동일 회사의 제품인 Power Compiler를 수행할 수 있어 전력 소모량 분석을 할 수 있는 기능을 가진다. Modelsim은 많이 알려져 있는 시뮬레이터이며, 테스트벤치에 따라 DUT(Design Under Test)에 대한 시뮬레이션을 수행하며, 시뮬레이션 결과를 파형으로 나타내준다.

시뮬레이션 및 전력 소모량 추정 과정을 그림 1에 나타내었다. 전력 소모량 추정 과정은 상기 두 개의 툴을 함께 이용하여야 한다. 두 개의 툴 사이의 인터페이스 역할을 하는 것이 SAIF(Switching Activity Interchange Format)이다[5]. 이는 로직이 동작할 때 내부 스위칭 활동을 기록할 수 있는 포맷이며, 시뮬레이터 프로그램이 시뮬레이션을 수행할 때 스위칭 활동을 기록하였다가 Design Compiler에서 이를 바탕으로 전력 소모량을 추정할 수 있도록 한다.



(그림 1) 전력 소모량 추정 과정 흐름도

2.1. Foward SAIF 생성

시뮬레이션을 수행하면서 스위칭 활동을 기록하기 위해서 스위칭 활동을 모니터링할 부분에 대한 정보를 시뮬레이터에게 알려주어야 할 필요가 있다. 이를 위해 Degisn Compiler에서는 rtl2saif 라는 명령을 제공하며, 이를 이용해 Forward SAIF 파일을 생성한다.

2.2. 시뮬레이션 및 스위칭 활동 기록

Modelsim에서는 다른 프로그램 언어로 작성된 루틴을 시뮬레이션시 실행할 수 있도록 PLI(Programming Language Interface)를 제공한다. 또한 Design Compiler에서는 스위칭 활동을 모니터링 하여 Backward SAIF 파일로 기록할 수 있는 루틴을 라이브러리로 제공하고 있다.

시뮬레이션을 위해서는 그림 1에 나타난 것과 같이 4가지의 입력이 필요하다. 먼저 Design Compiler 상에서 합성된 Gate-level netlist가 필요하며, 이를 가지고 시뮬레이션을 수행할 Testbench 파일이 필요하다. 또한 Forward SAIF 파일이 필요하며, 반도체 공정사에서 제공하는 각 Cell에 대한 verilog 파일이 필요하며, 이는 Gate-level netlist가 공정사에서 제공하는 라이브러리를 통해 합성이 되었기 때문에, 해당 Gate에 대한 정보가 있어야 시뮬레이션이 가능하므로 시뮬레이션시 필요하다. Testbench의 경우 위에서 언급한 PLI 루틴을 이용해 Forward SAIF 파일을 읽어와 시뮬레이션시 스위칭 활동을 모니터링 하여 Backward SAIF 파일로 출력하는 루틴이 작성되어 있어야 한다.

위의 입력들을 이용하여 시뮬레이션을 수행하면 Backward SAIF 파일이 생성되며, 이 파일은 각 모니터링 지점에 대해 TC(Toggle Count), T1(출력이 1인 시간), T0(출력이 0인 시간) 등이 기록되어 있어 전력 소모량 추정에 사용될 수 있다.

2.3. 전력 추정량 리포트 생성

시뮬레이션이 끝나면 다시 Design Compiler로 돌아와서 read_saif 명령을 통해 시뮬레이션 과정에서 생성된 Backward SAIF 파일을 읽어온다. 이후 report_power 라는 명령을 통해 전력 소모량에 관한 리포트를 확인할 수 있다. 리포트에서는 Net Switching Power, Cell Internal Power, Cell Leakage Power를 확인할 수 있다.

3. 실험에 사용된 파라메터

실험에 사용된 14개 SHA-3 후보에 대한 하드웨어 구현은 RCIS의 SHA-3 Hardware Project에서 사용된 RTL 코드를 사용하였으며, RCIS 홈페이지에 공개되어 있다[4]. 14개의 후보는 모두 256비트의 해쉬를 출력하도록 구현되어 있다. 비교를 위해 14개의 후보 외 추가로 SHA-256의 구현에 대해서도 실험을 진행하였다.

각 후보를 Gate-level netlist로 합성하기 위해 목표 동작 주파수를 10MHz로 설정하였으며, Testbench를 작성할 때도 위와 같은 동작 주파수로 동작하도록 하였다. 전력 소모 특성을 제대로 관찰하기 위해서는 충분히 긴 시간동안 시뮬레이션을 진행하여야 하지만, 시뮬레이션 시간이 오래 걸리는 것을 감안하여, 해쉬 모듈에 1000바이트 가량의 입력에 대한 해쉬 결과값이 나올 때 까지의 스위칭 활동을 기록하였다.

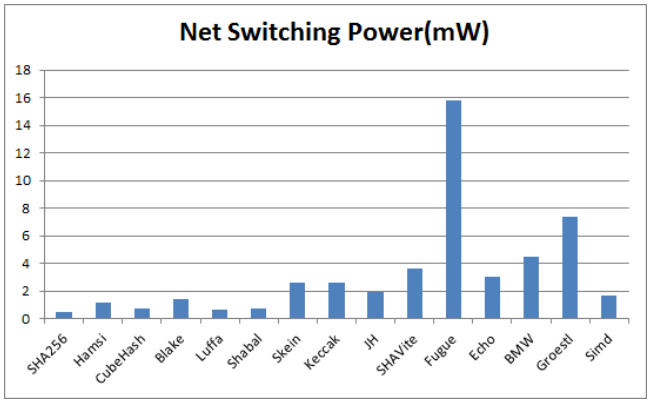
Gate-level netlist 합성을 위해 Magnachip/Hynix사의 0.18um 공정 라이브러리를 사용하였다.

4. 실험 결과

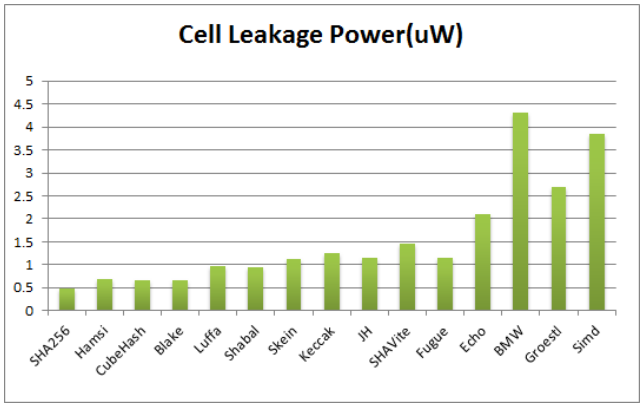
표 1은 SHA-3 2 라운드 후보들에 대한 전력 소모량 추정치 및 합성 후 칩의 면적을 보여주고 있다. (칩의 면적은 2 Input NAND gate의 개수로 환산한 값이다.) 모든 SHA-3 2 라운드 후보들이 SHA-256의 구현보다 Net Switching Power와 Cell Leakage Power에서 많은 전력을 소모하는 것으로 나타났다. 그림 2는 칩면적에 대해 오름차순으로 정리한 Net Switching Power의 그래프이며, 그림 3은 같은 방법으로 Cell Leakage Power를 나타낸 그래프이다. 그래프는 칩의 Cell Leakage Power가 칩의 사이즈와 어느 정도 연관성이 있다는 것을 보여주고 있다. Net Switching Power의 경우도 어느 정도의 연관성은 보

<표 2> SHA-3 2 Round 후보들에 대한 전력 소모량 추정 결과

SHA3-Candidates	Net Switching Power(mW)	Cell Leakage Power(uW)	Area (μm^2/NAND2X1)
Blake	1.4106	0.6561	30060.6181
BMW	4.488	4.3176	130875.381
CubeHash	0.7273	0.6645	24319.07268
Echo	3.045	2.0993	108627.1403
Fugue	15.7876	1.1398	71064.12039
Groestl	7.3994	2.7025	135353.7718
Hamsi	1.175	0.6872	22664.49393
JH	1.9503	1.1529	54026.68753
Keccak	2.6504	1.2405	44460.59742
Luffa	0.6231	0.9578	33451.40201
SHA256	0.4526	0.4841	19230.98804
Shabal	0.7287	0.9396	36857.45109
SHAVite	3.6129	1.4534	55833.25691
Simd	1.6709	3.8543	141620.7195
Skein	2.6131	1.1115	40597.97653



(그림 2) Net Switching Power 그래프 (칩면적 오름차순)



(그림 3) Cell Leakage Power 그래프 (칩면적 오름차순)

이지만, Cell Leakage Power에 비해서는 불규칙 적인 분포를 보인다. 이는 Net Switching Power는 칩의 면적에도 영향을 받지만, 해쉬 모듈이 동작할 때 얼마나 많은 스위칭 활동이 일어나는지가 전력 소모량 추정치에 영향을 미치기 때문이라고 추측된다.

5. 결론

본 논문에서는 H/W로 구현된 설계에 대해 시뮬레이션을 통해 전력 소모량을 추정하는 방법을 소개하였으며, SHA-3의 2 라운드 후보들에 대해서 H/W로 구현되었을 때의 전력 소모량 추정치를 도출해 내었다. 실험 결과 Luffa, CubeHash, Shabal 등이 스위칭 전력 소모에 있어 뛰어난 성능을 보임을 확인할 수 있었으며, 실험 결과에 대한 원인은 차후 연구과제로 남긴다.

참고문헌

[1] National Institute of Standards and Technology (NIST), “Cryptographic Hash Algorithm Competition”, http://csrc.nist.gov/group/ST/hash/sha-3/index.html

[2] X. Wang, A. C. Yao and F. Yao, “Cryptanalysis on SHA-1”, CRYPTOGRAPHIC HASH WORKSHOP, Oct 31-Nov 1, 2005.

[3] X. Wang, Y. L. Yin and H. Yu, “Finding Collisions in the Full SHA-1”, Crypto 2005, LNCS 3621, pp. 17-36, 2005.

[4] Research Center for Information Security(RCIS), “SHA-3 Hardware Project”, http://www.rcis.aist.go.jp/special/SASEBO/SHA3-en.html

[5] Synopsys, "Power Compiler User Guide", https://solvnet.synopsys.com/dow_retrieve/E-2010.09/ni/power.html#Power%20Compiler