(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2024-0125273
(43) 공개일자 2024년08월19일

(51) 국제특허분류(Int. Cl.)
G06F 21/76 (2023.01) *G06F 21/14* (2013.01)
(52) CPC특허분류
G06F 21/76 (2023.02)
G06F 21/14 (2013.01)
(21) 출원번호 10-2023-0018130
(22) 출원일자 2023년02월10일
심사청구일자 2023년02월10일

(71) 출원인
국방과학연구소
대전광역시 유성구 북유성대로488번길 160 (수남동)
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
조만희
서울특별시 성북구 안암로 145 고려대학교 신공학관 427호
이동찬
서울특별시 마포구 와우산로 94 홍익대학교 제2공학관 412호
(뒷면에 계속)
(74) 대리인
특허법인 광장리앤고

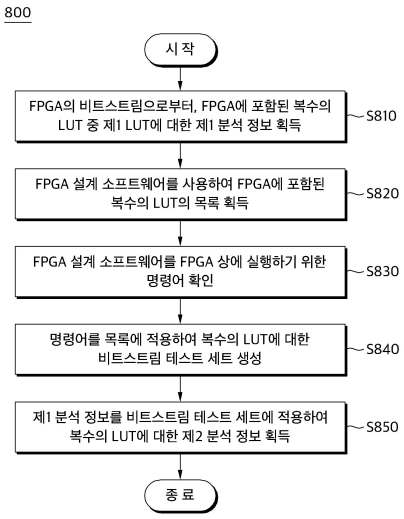
전체 청구항 수 : 총 11 항

(54) 발명의 명칭 **역공학을 사용하여 FPGA를 분석하는 장치 및 그의 동작 방법**

(57) 요 약

본 발명은 전자 장치가 FPGA를 분석하는 방법으로서, 상기 FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득하는 단계; FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하는 단계; 상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하는 단계; 상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하는 단계; 및 상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법에 관한 것이다.

대 표 도 - 도8



(72) 발명자

이상현

서울특별시 마포구 와우산로 94 홍익대학교 제2공학관 412호

김영민

서울특별시 마포구 와우산로 94 홍익대학교 제2공학관 708호

이형민

서울특별시 성북구 안암로 145 고려대학교 신공학관 516호

명 세 서

청구범위

청구항 1

전자 장치가 FPGA(Field-Programmable Gate Array)를 분석하는 방법으로서,

상기 FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT(Look Up Table) 중 제1 LUT에 대한 제1 분석 정보를 획득하는 단계;

FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하는 단계;

상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하는 단계;

상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하는 단계; 및

상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 2

제1항에 있어서,

상기 제1 분석 정보는 상기 제1 LUT에 대한 비트스트림 변환 규칙 정보 및 상기 제1 LUT의 비트스트림 상 주소 정보를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 3

제1항에 있어서,

상기 제1 분석 정보를 획득하는 단계는

상기 FPGA의 상기 비트스트림을 추출하는 단계;

상기 비트스트림으로부터 상기 FPGA의 PLP(Programmable Logic Points) 정보를 추출하는 단계; 및

상기 PLP 정보를 기반으로 상기 제1 LUT에 대한 역공학을 수행하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 4

제3항에 있어서,

상기 비트스트림을 추출하는 단계는

전원이 상기 FPGA에 인가된 후 메모리로부터 상기 FPGA로 다운로드되는 상기 비트스트림을 추출하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 5

제1항에 있어서,

상기 복수의 LUT의 목록은 상기 FPGA에 포함된 CLB(Configurable Logic Block) 타일 구조에 대한 정보, 및 상기 CLB 타일 내 포함된 LUT에 대한 정보를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 6

제1항에 있어서,

상기 명령어는 FPGA 보드 설정 및 비트스트림 생성을 위한 제1 명령어와 회로 배치 및 배선 설정을 위한 제2 명령어 중 적어도 하나를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 7

제1항에 있어서,

상기 비트스트림 테스트 세트는 상기 제1 분석 정보에 대응하는 논리 회로를 상기 복수의 LUT 중 상기 제1 LUT와 상이한 LUT에 배치하도록 생성되는, 전자 장치가 FPGA를 분석하는 방법.

청구항 8

제1항에 있어서,

상기 제2 분석 정보는 상기 복수의 LUT의 상기 비트스트림 상 주소 정보를 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 9

제8항에 있어서,

상기 복수의 LUT의 상기 비트스트림 상 주소 정보를 기반으로 상기 비트스트림을 분리하는 단계; 및

상기 제1 분석 정보에 포함된 비트스트림 변환 규칙을 기반으로 상기 복수의 LUT 각각에 구현된 논리 회로를 획득하는 단계를 더 포함하는, 전자 장치가 FPGA를 분석하는 방법.

청구항 10

비일시적 컴퓨터 판독 가능 저장 매체로서,

컴퓨터 판독 가능 명령어들을 저장하도록 구성되는 매체를 포함하고,

상기 컴퓨터 판독 가능 명령어들은 프로세서에 의해 실행되는 경우 상기 프로세서가:

FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득하는 단계;

FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하는 단계;

상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하는 단계;

상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하는 단계; 및

상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법을 수행하도록 하는, 비일시적 컴퓨터 판독 가능 저장 매체.

청구항 11

FPGA를 분석하는 전자 장치로서,

적어도 하나의 명령어를 저장하는 메모리; 및

상기 적어도 하나의 명령어를 실행함으로써, 상기 FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득하고, FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하고, 상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하고, 상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하고, 상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 프로세서를 포함하는, FPGA를 분석하는 전자 장치.

발명의 설명

기 술 분 야

[0001] 본 명세서의 실시 예는 역공학(reverse engineering)을 사용하여 FPGA(Field-Programmable Gate Array)를 분석하는 장치 및 그의 동작 방법에 관한 것이다.

배 경 기 술

[0003] 기존 하드웨어 역공학은 반도체 집적 회로(integrated circuit, IC)를 주 대상으로 하여 진행되어 왔으나, 최근 4차 산업혁명과 이를 대표하는 IoT(Internet of Things) 산업의 핵심 부품으로 FPGA가 사용되기 시작하면서 FPGA 역공학의 중요성 또한 크게 증가하게 되었다. FPGA는 상용 임베디드 시스템 제품에 많이 사용되고 있으며, 시스템 개발의 편의성, 재설계 가능함의 장점 외에도 준수한 성능으로 인해 그 수요량이 기하급수적으로 증가하고 있다. FPGA는 고속 병렬 데이터 처리가 가능해 임베디드 시스템뿐만 아니라 영상 처리 및 최신 AI 기술 제품에 매우 적합하며, 사용 분야의 특성상 FPGA의 보안/역공학 기술은 산업, 국방 분야에서 큰 중요성을 가진다.

[0004] 일반적으로 역공학은 기존에 존재하는 시스템을 분석하고 구성 컴포턴트와 동일한 연관성을 가지는 카피를 생성하는 작업을 말한다. FPGA에서의 역공학은 다양한 방법으로 FPGA 프로그래밍 파일인 넷리스트를 재구성하는 것을 지칭할 수 있다.

[0005] FPGA 보안/역공학에 관한 한 가지 예를 살펴보면, 국방, 범죄수사, 첩보전 등의 분야에서 취득한 하드웨어 장비들을 분석하여 그 목적과 기능, 저장된 데이터를 찾아내는 기법이 요구된다. 또한, 이러한 하드웨어 전자 장비들은 FPGA를 이용하여 쉽게 시스템으로 구현될 수 있다. 구체적인 예로 2014년 카메라 촬영이 가능한 드론이 군사지역을 순회하다가 추락하는 사건이 있었는데, 그 드론 하드웨어를 분석하여 드론의 이동 경로 및 저장 데이터를 분석할 수 있었으며, 이는 국방 보안에 큰 기여를 하였다. 당시 포획된 드론의 하드웨어 구현은 간단한 수준이어서 어렵지 않게 분석 및 역공학이 가능했으나, 앞으로는 보다 정밀하고 암호화된 하드웨어에 대한 역공학 기술이 필요하다는 것을 알 수 있다.

[0006] 또한, 한편, 반도체 칩의 경우 내부 회로가 물리적으로 결정되어 있기 때문에 반도체 칩의 상단을 황산이나 사포 등을 이용하여 제거한 후 반도체 칩을 노출시킨다면 직접 물리적으로 역공학을 수행하는 것이 가능하다. 하지만 이와 같은 침습적인(invasive) 방법을 사용하여 역공학을 시도한다면 그 반도체 칩은 더 이상 사용할 수 없으며 다시 역공학을 시도하는 것도 불가능할 것이다. 따라서 획득한 하드웨어의 수가 적은 경우에는 침습적인 역공학 방법을 시도하기 어렵다. 또한 침습적인 방법은 지금까지 여러 반도체 칩에 시도되어 왔기 때문에 그에 대한 물리적인 대책(예를 들어, 더미 레이어(dummy layer)로 회로를 덮어 파악이 어렵게 함) 또한 많이 개발되었다. 따라서 비침습적인(non-invasive) 방법을 사용한 역공학의 필요성이 더욱 커지고 있다.

[0007] 나아가, 선행 FPGA 역공학 기법에서는 개별 LUT에 대한 비트스트림 변환 규칙과 비트스트림 상 주소 값을 획득함으로써 논리 회로를 추측할 수 있으나, FPGA 내부에 포함된 대량의 LUT를 분석하기에는 한계가 있다. 따라서, 기존 역공학 기법을 기반으로 대량의 LUT 분석을 자동화하는 방법이 요구된다.

[0008] 관련된 선행기술문헌으로는 대한민국공개특허공보 제10-2016-0063990호(2016년 6월 7일 공개)가 있다.

발명의 내용

해결하려는 과제

[0010] 본 명세서의 실시 예는 상술한 문제점을 해결하기 위하여 제안된 것으로, FPGA 내부에 구성된 논리 회로 역공학을 위한 비트스트림(bitstream) 자동 분석 기법을 제공하는 것을 목적으로 한다. 보다 구체적으로, 본 명세서의 실시 예는 FPGA 구조 정보를 포함하는 데이터시트와 FPGA 설계 소프트웨어 도구에서 제공되는 정보를 활용하여 FPGA 내부 모든 LUT에 대한 비트스트림 정보를 획득함으로써 역공학 범위를 크게 확장하는 것을 목적으로 한다.

과제의 해결 수단

[0012] 상술한 과제를 달성하기 위하여, 본 명세서의 일 실시 예에 따르는 전자 장치가 FPGA(Field-Programmable Gate Array)를 분석하는 방법은, 상기 FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT(Look Up Table) 중 제1 LUT에 대한 제1 분석 정보를 획득하는 단계; FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하는 단계; 상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하는 단계; 상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하는 단계; 및 상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 단계를 포함하는 것을 특징으로 한다.

[0013] 일 실시 예에 따르면, 상기 제1 분석 정보는 상기 제1 LUT에 대한 비트스트림 변환 규칙 정보 및 상기 제1 LUT의 비트스트림 상 주소 정보를 포함하는 것을 특징으로 한다.

[0014] 일 실시 예에 따르면, 상기 제1 분석 정보를 획득하는 단계는 상기 FPGA의 상기 비트스트림을 추출하는 단계; 상기 비트스트림으로부터 상기 FPGA의 PLP(Programmable Logic Points) 정보를 추출하는 단계; 및 상기 PLP 정보를 기반으로 상기 제1 LUT에 대한 역공학을 수행하는 단계를 포함하는 것을 특징으로 한다.

[0015] 일 실시 예에 따르면, 상기 비트스트림을 추출하는 단계는 전원이 상기 FPGA에 인가된 후 메모리로부터 상기 FPGA로 다운로드되는 상기 비트스트림을 추출하는 단계를 포함하는 것을 특징으로 한다.

[0016] 일 실시 예에 따르면, 상기 복수의 LUT의 목록은 상기 FPGA에 포함된 CLB(Configurable Logic Block) 타일 구조에 대한 정보, 및 상기 CLB 타일 내 포함된 LUT에 대한 정보를 포함하는 것을 특징으로 한다.

[0017] 일 실시 예에 따르면, 상기 명령어는 FPGA 보드 설정 및 비트스트림 생성을 위한 제1 명령어와 회로 배치 및 배선 설정을 위한 제2 명령어 중 적어도 하나를 포함하는 것을 특징으로 한다.

[0018] 일 실시 예에 따르면, 상기 비트스트림 테스트 세트는 상기 제1 분석 정보에 대응하는 논리 회로를 상기 복수의 LUT 중 상기 제1 LUT와 상이한 LUT에 배치하도록 생성되는 것을 특징으로 한다.

[0019] 일 실시 예에 따르면, 상기 제2 분석 정보는 상기 복수의 LUT의 상기 비트스트림 상 주소 정보를 포함하는 것을 특징으로 한다.

[0020] 일 실시 예에 따르면, 상기 전자 장치가 FPGA를 분석하는 방법은 상기 복수의 LUT의 상기 비트스트림 상 주소 정보를 기반으로 상기 비트스트림을 분리하는 단계; 및 상기 제1 분석 정보에 포함된 비트스트림 변환 규칙을 기반으로 상기 복수의 LUT 각각에 구현된 논리 회로를 획득하는 단계를 더 포함하는 것을 특징으로 한다.

[0021] 본 명세서의 일 실시 예에 따르는 비일시적 컴퓨터 판독 가능 저장 매체는, 컴퓨터 판독 가능 명령어들을 저장하도록 구성되는 매체를 포함하고, 상기 컴퓨터 판독 가능 명령어들은 프로세서에 의해 실행되는 경우 상기 프로세서가: FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득하는 단계; FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하는 단계; 상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하는 단계; 상기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하는 단계; 및 상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 단계를 포함하는, 전자 장치가 FPGA를 분석하는 방법을 수행하도록 하는 것을 특징으로 한다.

[0022] 본 명세서의 일 실시 예에 따르는 FPGA를 분석하는 전자 장치는, 적어도 하나의 명령어를 저장하는 메모리; 및 상기 적어도 하나의 명령어를 실행함으로써, 상기 FPGA의 비트스트림으로부터, 상기 FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득하고, FPGA 설계 소프트웨어를 사용하여 상기 FPGA에 포함된 상기 복수의 LUT의 목록을 획득하고, 상기 FPGA 설계 소프트웨어를 상기 FPGA 상에 실행하기 위한 명령어를 확인하고, 상

기 명령어를 상기 목록에 적용하여 상기 복수의 LUT에 대한 비트스트림 테스트 세트를 생성하고, 상기 제1 분석 정보를 상기 비트스트림 테스트 세트에 적용하여 상기 복수의 LUT에 대한 제2 분석 정보를 획득하는 프로세서를 포함하는 것을 특징으로 한다.

발명의 효과

[0024] 본 명세서의 실시 예에 따르면, 기존 FPGA PLP 역공학 방식이 한 개의 LUT를 분석하여 구현된 논리 회로를 복원하는 방식인 것으로부터 더 나아가, FPGA 내 모든 LUT를 활용하는 알고리즘을 구현하여 모든 LUT에 대한 분석 결과를 획득할 수 있다. 또한, 본 발명을 통해 전자 장치는 공통된 변환 규칙으로 구현되는 논리 회로의 테스트 세트를 자동으로 생성하고 이를 분석할 수 있다.

[0025] 또한, 본 명세서의 실시 예에 따르는 분석 방법은 FPGA의 구조 분석을 기반으로 하여 최신 FPGA 뿐만 아니라 구형 FPGA에도 적용 가능하여 범용성이 높다. 최근 출시되는 사물 인터넷(IoT) 기기들과 임베디드 시스템에는 다양한 FPGA가 사용되고 있는데, FPGA를 대상으로 한 하드웨어 공격에 대한 우려 또한 증가하고 있다. FPGA 역공학은 범죄 예방 및 하드웨어 보안 문제 해결에 큰 도움이 될 것이며, 비침습적인 역공학은 그 외 수사, 증거 수집, 군사 방어 등 대상 하드웨어를 손상시키지 않아야 하는 다양한 분야에 널리 사용될 수 있다.

[0026] 또한, 본 명세서의 실시 예에 따르는 분석 방법은 국방 보안에 큰 기여를 할 수 있다. FPGA 칩은 재프로그래밍이 가능한 특성상 prototyping으로 많이 사용되며 소량으로 제작할 때 주문형 반도체 칩에 비해 가격 대비 성능비가 좋기 때문에 다양한 기능의 소량 제품 제작에 많이 사용될 수 있다. 국방 분야의 정찰, 보안, 자동화, 무인화 등에 사용되는 많은 제품들이 FPGA를 사용하여 제작될 수 있으므로, FPGA 역공학 및 분석 기술을 보유한다면 군사 보안 및 정보전 분야에서 큰 우위를 점할 수 있을 것이다.

도면의 간단한 설명

[0028] 도 1은 본 명세서의 일 실시 예에 따른 전자 장치의 각 구성을 개략적으로 도시하는 예시적인 도면이다.

도 2는 본 명세서의 일 실시 예에 따른 비트스트림 구조 자동 분석 방법의 흐름을 나타내는 순서도이다.

도 3은 본 명세서의 일 실시 예에 따른 FPGA 내 CLB 타일 구조 및 목록을 나타내는 예시적인 도면이다.

도 4는 본 명세서의 일 실시 예에 따른 설계 소프트웨어 구동 명령어를 나타내는 예시적인 도면이다.

도 5는 본 명세서의 일 실시 예에 따른 서로 다른 LUT에 배치된 논리 회로에 대응하는 비트스트림 및 비교표를 예시적으로 나타내는 도면이다.

도 6은 본 명세서의 일 실시 예에 따른 비트스트림 테스트 세트를 자동으로 생성 및 분석하는 방법의 흐름을 나타내는 순서도이다.

도 7은 본 명세서의 일 실시 예에 따른 자동화 기법을 통한 FPGA 내 논리 회로 구성의 역공학 과정을 예시적으로 나타내는 도면이다.

도 8은 본 명세서의 일 실시 예에 따른 전자 장치가 FPGA를 분석하는 방법의 흐름을 나타내는 순서도이다.

발명을 실시하기 위한 구체적인 내용

[0029] 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0030] 실시 예를 설명함에 있어서 본 발명이 속하는 기술 분야에 익히 알려져 있고 본 발명과 직접적으로 관련이 없는 기술 내용에 대해서는 설명을 생략한다. 이는 불필요한 설명을 생략함으로써 본 발명의 요지를 흐리지 않고 더욱 명확히 전달하기 위함이다.

[0031] 마찬가지 이유로 첨부 도면에 있어서 일부 구성요소는 과장되거나 생략되거나 개략적으로 도시되었다. 또한, 각 구성요소의 크기는 실제 크기를 전적으로 반영하는 것이 아니다. 각 도면에서 동일한 또는 대응하는 구성요소에는 동일한 참조 번호를 부여하였다.

[0032] 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시

예들을 참조하면 명확해질 것이다. 그러나 본 발명은 이하에서 개시되는 실시 예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시 예들은 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다. 명세서 전체에 걸쳐 동일 참조 부호는 동일 구성 요소를 지칭한다.

[0033] 이 때, 처리 흐름도 도면들의 각 블록과 흐름도 도면들의 조합들은 컴퓨터 프로그램 인스트럭션들에 의해 수행될 수 있음을 이해할 수 있을 것이다. 이들 컴퓨터 프로그램 인스트럭션들은 범용 컴퓨터, 특수용 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비의 프로세서에 탑재될 수 있으므로, 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비의 프로세서를 통해 수행되는 그 인스트럭션들이 흐름도 블록(들)에서 설명된 기능들을 수행하는 수단을 생성하게 된다. 이들 컴퓨터 프로그램 인스트럭션들은 특정 방식으로 기능을 구현하기 위해 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비를 지향할 수 있는 컴퓨터 이용 가능 또는 컴퓨터 판독 가능 메모리에 저장되는 것도 가능하므로, 그 컴퓨터 이용가능 또는 컴퓨터 판독 가능 메모리에 저장된 인스트럭션들은 흐름도 블록(들)에서 설명된 기능을 수행하는 인스트럭션 수단을 내포하는 제조 품목을 생산하는 것도 가능하다. 컴퓨터 프로그램 인스트럭션들은 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비 상에 탑재되는 것도 가능하므로, 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비 상에서 일련의 동작 단계들이 수행되어 컴퓨터로 실행되는 프로세스를 생성해서 컴퓨터 또는 기타 프로그램 가능한 데이터 프로세싱 장비를 수행하는 인스트럭션들은 흐름도 블록(들)에서 설명된 기능들을 실행하기 위한 단계들을 제공하는 것도 가능하다.

[0034] 또한, 각 블록은 특정된 논리적 기능(들)을 실행하기 위한 하나 이상의 실행 가능한 인스트럭션들을 포함하는 모듈, 세그먼트 또는 코드의 일부를 나타낼 수 있다. 또, 몇 가지 대체 실행 예들에서는 블록들에서 언급된 기능들이 순서를 벗어나서 발생하는 것도 가능함을 주목해야 한다. 예컨대, 잇달아 도시되어 있는 두 개의 블록들은 사실 실질적으로 동시에 수행되는 것도 가능하고 또는 그 블록들이 때때로 해당하는 기능에 따라 역순으로 수행되는 것도 가능하다.

[0035] 이 때, 본 실시 예에서 사용되는 '~부'라는 용어는 소프트웨어 또는 FPGA또는 ASIC과 같은 하드웨어 구성요소를 의미하며, '~부'는 어떤 역할들을 수행한다. 그렇지만 '~부'는 소프트웨어 또는 하드웨어에 한정되는 의미는 아니다. '~부'는 어드레싱할 수 있는 저장 매체에 있도록 구성될 수도 있고 하나 또는 그 이상의 프로세서들을 재생시키도록 구성될 수도 있다. 따라서, 일 예로서 '~부'는 소프트웨어 구성요소들, 객체지향 소프트웨어 구성요소들, 클래스 구성요소들 및 태스크 구성요소들과 같은 구성요소들과, 프로세스들, 함수들, 속성들, 프로시저들, 서브루틴들, 프로그램 코드의 세그먼트들, 드라이버들, 펌웨어, 마이크로코드, 회로, 데이터, 데이터베이스, 데이터 구조들, 테이블들, 어레이들, 및 변수들을 포함한다. 구성요소들과 '~부'들 안에서 제공되는 기능은 더 작은 수의 구성요소들 및 '~부'들로 결합되거나 추가적인 구성요소들과 '~부'들로 더 분리될 수 있다. 뿐만 아니라, 구성요소들 및 '~부'들은 디바이스 또는 보안 멀티미디어카드 내의 하나 또는 그 이상의 CPU들을 재생시키도록 구현될 수도 있다.

[0037] 도 1은 본 명세서의 일 실시 예에 따른 전자 장치의 각 구성을 개략적으로 도시하는 예시적인 도면이다.

[0038] 도 1을 참조하면, 전자 장치(100)는 프로세서(110) 및 메모리(120)를 포함할 수 있고, 역공학을 사용하여 FPGA를 분석하는 방법을 수행할 수 있다. 도 1에 도시된 전자 장치(100)에는 본 실시 예들과 관련된 구성요소들만이 도시되어 있다. 따라서, 전자 장치(100)에는 도 1에 도시된 구성요소들 외에 다른 범용적인 구성요소들이 더 포함될 수 있음은 당해 기술분야의 통상의 기술자에게 자명하다.

[0039] 프로세서(110)는 전자 장치(100)에서의 FPGA 분석을 위한 전반적인 기능들을 제어하는 역할을 한다. 예를 들어, 프로세서(110)는 전자 장치(100) 내의 메모리(120)에 저장된 프로그램들을 실행함으로써, 전자 장치(100)를 전반적으로 제어한다. 프로세서(110)는 전자 장치(100) 내에 구비된 CPU(central processing unit), GPU(graphics processing unit), AP(application processor) 등으로 구현될 수 있으나, 이에 제한되지 않는다.

[0040] 메모리(120)는 전자 장치(100) 내에서 처리되는 각종 데이터들을 저장하는 하드웨어로서, 메모리(120)는 전자 장치(100)에서 처리된 데이터들 및 처리될 데이터들을 저장할 수 있다. 또한, 메모리(120)는 전자 장치(100)에 의해 구동될 애플리케이션들, 드라이버들 등을 저장할 수 있다. 메모리(120)는 DRAM(dynamic random access memory), SRAM(static random access memory) 등과 같은 RAM(random access memory), ROM(read-only memory),

EEPROM(electrically erasable programmable read-only memory), CD-ROM, 블루레이 또는 다른 광학 디스크 스토리지, HDD(hard disk drive), SSD(solid state drive), 또는 플래시 메모리를 포함할 수 있다.

[0042] 도 2는 본 명세서의 일 실시 예에 따른 비트스트림 구조 자동 분석 방법의 흐름을 나타내는 순서도이다.

[0043] 도 2를 참조하면, 본 명세서의 일 실시 예에 따른 FPGA 내부 논리 회로 역공학을 위한 비트스트림 구조 자동 분석 방법의 흐름도(200)이 도시된다. 일 실시 예에서, 도 2에 도시된 방법은 도 1에 도시된 전자 장치(100)에 의해 수행될 수 있다.

[0044] 일 실시 예에서, 전자 장치(100)는 단일 LUT에 대한 PLP(Programmable Logic Points) 역공학 기법(210)을 수행하여 단일 LUT에 대한 비트스트림 변환 규칙과 데이터 주소 정보와 같은 분석 정보를 획득할 수 있다.

[0045] 일 실시 예에서, 전자 장치(100)는 PLP 역공학(210)을 사용하여 획득한 단일 LUT에 대한 분석 정보를 기반으로 FPGA 내 모든 LUT에 대한 분석 정보를 획득하기 위해 역공학 자동화 기법(220)을 수행할 수 있다. 전자 장치(100)는 FPGA 설계 소프트웨어 도구(221)에서 제공하는 정보를 이용할 수 있다. 전자 장치(100)는 FPGA 내부 모든 구성요소들로부터 역공학 가능한 구성 회로들의 목록(222)을 획득하고, 다양한 구성 회로들의 논리식 설계와 활성화 여부를 제어하는 소프트웨어 구동 명령어(223)를 확인할 수 있다. 그 다음, 전자 장치(100)는 FPGA의 비트스트림 자동 생성 알고리즘(224) 및 비트스트림 자동 분석 알고리즘(225)을 작성할 수 있다. 전자 장치(100)는 FPGA 구성 회로 목록(222)과 소프트웨어 구동 명령어(223) 제어를 통해 모든 LUT에 대한 비트스트림 테스트 세트를 생성할 수 있고, 기존에 PLP 역공학 기법(210)을 통해 획득한 단일 LUT에 대한 비트스트림 변환 규칙과 주소 정보를 기반으로 테스트 세트를 분석하여 최종적으로 FPGA 내 모든 LUT에 대한 비트스트림 분석 정보를 획득할 수 있다.

[0046] 일 실시 예에서, 전자 장치(100)는 FPGA의 비트스트림으로부터, FPGA에 포함된 복수의 LUT(Look Up Table) 중 제1 LUT에 대한 제1 분석 정보를 획득할 수 있다. 일 실시 예에서, 제1 분석 정보는 제1 LUT에 대한 비트스트림 변환 규칙 정보 및 제1 LUT의 비트스트림 상 주소 정보를 포함할 수 있다. 전자 장치(100)는 FPGA에 포함된 단일 LUT에 대한 역공학 방법(210)을 수행함으로써 단일 LUT에 대한 비트스트림 변환 규칙과 데이터 주소 정보를 포함하는 분석 정보를 획득할 수 있다.

[0047] 한편, 반도체 칩의 경우 내부 회로가 물리적으로 결정되어 있기 때문에 반도체 칩의 상단을 황산이나 사포 등을 이용하여 제거한 후 반도체 칩을 노출시킨다면 직접 물리적으로 역공학을 수행하는 것이 가능하다. 하지만 이와 같은 침습적인(invasive) 방법을 사용하여 역공학을 시도한다면 그 반도체 칩은 더 이상 사용할 수 없으며 다시 역공학을 시도하는 것도 불가능할 것이다. 따라서 획득한 하드웨어의 수가 적은 경우에는 침습적인 역공학 방법을 시도하기 어렵다. 또한 침습적인 방법은 지금까지 여러 반도체 칩에 시도되어 왔기 때문에 그에 대한 물리적인 대책(예를 들어, 더미 레이어(dummy layer)로 회로를 덮어 파악이 어렵게 함) 또한 많이 개발되었다. 따라서 비침습적인(non-invasive) 방법을 사용한 역공학의 필요성이 더욱 커지고 있다.

[0048] FPGA 내부에는 저비용, 재프로그래밍, 속도 등의 이유로 SRAM(static random access memory)이 주로 사용되는데, SRAM은 휘발성 메모리이므로 FPGA 프로그래밍 데이터를 저장하기 위해서는 외부에 추가적인 비휘발성 메모리가 필요하다. FPGA는 외부 PROM 또는 플래시 메모리에 프로그래밍 데이터인 비트스트림을 저장하여 보관할 수 있으며, 전원이 인가된 후 비트스트림을 통해서 FPGA의 내부 논리 회로를 구성할 수 있다. FPGA에 전원을 인가하면 외부 비휘발성 메모리(예를 들어, PROM(programmable read only memory))에서 FPGA로 프로그래밍 신호가 다운로드되며, 이때 다운로드 경로에서 추출한 비트스트림을 활용하여 FPGA 역공학을 수행할 수 있다. 일 실시 예에서, 비트스트림은 논리 분석기(logic analyzer)와 같은 신호 측정 장비를 통해 추출될 수 있으며, 이를 통해 FPGA를 손상시키지 않는 비침습적 역공학이 수행될 수 있다. 비트스트림 내 데이터는 FPGA 내부의 논리 회로에 대응할 수 있으며, 데이터의 변환 규칙과 각 주소별 대응하는 회로를 분석하여 비트스트림을 통해 설계되는 내부 논리 회로를 추측하는 것이 가능할 수 있다.

[0049] FPGA 비트스트림 역공학은 PIP(Programmable Interconnect Points) 및 PLP(Programmable Logic Points) 추출로 구분될 수 있다. PIP는 프로그래밍 가능한 CMOS 트랜지스터 스위치를 사용하여 IOB(Input/Output Block), CLB(Configurable Logic Block) 및 다양한 타일을 서로 연결하는 데 사용되며 상호연결(interconnect, INT) 타일 내부에 주로 구성될 수 있다. PLP는 CLB 내부의 논리 회로를 구현하는 데 사용될 수 있다. CLB 내부는 LUT, 멀티플렉서(MUX), 캐리(Carry) 등으로 구성될 수 있으며, LUT의 입력 조합에 따라 다양한 논리 회로가 구현될 수 있다.

[0050] FPGA 내부에는 동일한 구성 회로들을 가지는 대량의 타일들이 포함되어 있으며, 기본 논리 회로는 논리식의 복잡성에 따라 LUT 하나 또는 다수의 조합으로 구현될 수 있다. 동일한 구성 회로에 대해 비트스트림 상 데이터 크기는 동일하고, 유사한 비트스트림 변환 규칙을 가진다. 따라서, 단일 회로(예를 들어, 제1 LUT)에 대한 역공학을 통해 비트스트림 변환 규칙을 파악하면 이를 FPGA 내 동일한 회로에 대한 분석으로 확장할 수 있다.

[0051] 일 실시 예에서, 제1 LUT에 대한 분석 정보를 획득하는 것은, FPGA의 비트스트림을 추출하고, 비트스트림으로부터 FPGA의 PLP 정보를 추출하고, PLP 정보를 기반으로 제1 LUT에 대한 역공학을 수행하는 것을 포함할 수 있다. 일 실시 예에서, 전자 장치는 FPGA의 비트스트림을 추출하고, 비트스트림의 PLP 정보를 추출하고, PLP 정보를 이용하여 FPGA에 포함된 복수의 CLB 섹션과 복수의 INT 섹션을 매칭하여 2차원 형태의 구조 테이블을 생성하고, 복수의 INT 섹션에 포함된 복수의 PIP 정보를 추출하고, 복수의 PIP 정보를 이용하여 역공학 로직을 생성할 수 있다. 일 실시 예에서, 비트스트림을 추출하는 것은 전원이 FPGA에 인가된 후 메모리로부터 FPGA로 다운로드되는 비트스트림을 추출하는 것을 포함할 수 있다.

[0053] 도 3은 본 명세서의 일 실시 예에 따른 FPGA 내 CLB 타일 구조 및 목록을 나타내는 예시적인 도면이다. 도 3을 참조하면, FPGA 내 CLB 타일 구조(310) 및 CLB 타일 내 슬라이스 구조(320)와, CLB 타일 내 슬라이스의 목록(330)이 도시된다.

[0054] 일 실시 예에서, 전자 장치(100)는 FPGA 설계 소프트웨어 도구를 사용하여 FPGA에 포함된 복수의 LUT의 목록을 획득할 수 있다. 일 실시 예에서, 복수의 LUT의 목록은 FPGA에 포함된 CLB 타일 구조에 대한 정보, 및 CLB 타일 내 포함된 LUT에 대한 정보를 포함할 수 있다. 도 3은 Xilinx 사의 Vivado Design Suite 소프트웨어 도구를 사용하여 Xilinx 사의 Artix-7 제품군 FPGA의 내부를 확인한 예시이다. CLB 타일 구조(310)에서 FPGA의 CLB 타일 구조가 x축 및 y축 값으로 표현되며, CLB 타일 내부의 슬라이스에는 논리 회로를 구성하는 LUT가 포함되어 있다. 슬라이스 내 LUT는 슬라이스 구조(320)에 도시된 바와 같이 A, B, C 및 D의 4개의 명칭으로 세부 구분될 수 있으며, 4개의 LUT는 동일한 회로로 구성될 수 있다. 즉, 하나의 CLB는 2개의 슬라이스(예를 들어, L-Slice, R-Slice)로 구성될 수 있으며, 각 슬라이스는 4개의 LUT(A, B, C, D)를 포함할 수 있다. 여기에서 FPGA 설계 소프트웨어 도구를 사용하면 목록(330)과 같이 타일과 슬라이스의 목록이 획득될 수 있다. 목록(330)의 하위 항목으로 4개의 LUT를 포함시키면 FPGA 내부의 모든 LUT에 대한 목록이 생성될 수 있다.

[0055] Xilinx 사의 Artix-7 제품군은 2,600개의 CLB와 20,800의 LUT를 포함할 수 있으며, 헤더 파일을 제외한 구성 정보 비트스트림의 길이는 2,192,012 바이트일 수 있다. 비트스트림에서 하나의 LUT는 8바이트이며, 이 비트가 연속적으로 배치되지 않고 2바이트씩 4등분으로 분리되어 배치될 수 있다.

[0056] Xilinx 사의 FPGA 시리즈 제품군의 비트스트림 구조는 모두 동일하다. 비트스트림은 헤더(header), 커맨드 프레임(command frame), 구성 데이터(configuration data) 및 순환 중복 검사(cyclic redundancy check, CRC)로 분할될 수 있다. 헤더는 대상 FPGA의 칩 명칭, 프로젝트 명칭, 생성일자, 및 비트스트림 크기와 같은 정보를 포함할 수 있다. 커맨드 프레임은 프레임 주소 레지스터와 IDCODE와 같은 정보를 포함할 수 있으며, 이는 구성 데이터의 시작 위치를 나타낼 수 있다. 각 FPGA에는 고유한 IDCODE가 할당될 수 있으며, 이는 FPGA의 유형을 식별하는 데 사용될 수 있다. 구성 데이터는 FPGA 내부 회로의 구성을 결정할 수 있으며, 구성 데이터 값에 따라 다양한 회로가 FPGA 내에 설계될 수 있다. CRC는 PROM에 저장된 비트스트림이 FPGA로 성공적으로 다운로드 되었는지를 결정할 수 있다.

[0058] 도 4는 본 명세서의 일 실시 예에 따른 설계 소프트웨어 구동 명령어를 나타내는 예시적인 도면이다. 도 4를 참조하면, 예시적인 논리 회로 소스 코드(410), FPGA 보드 설정 및 비트스트림 생성 명령어(420) 및 회로 배치 및 배선 설정 명령어(430)가 도시된다.

[0059] 일 실시 예에서, 전자 장치(100)는 FPGA 설계 소프트웨어 도구를 FPGA 상에 실행하기 위한 소프트웨어 구동 명령어를 확인할 수 있다. 소프트웨어 구동 명령어를 사용하면 설계 회로의 세부 배치와 연결을 임의로 설정할 수 있으며, 파일 생성부터 비트스트림 생성까지의 동작을 자동으로 수행할 수 있다. 명령어(420)는 사용하고자 하는 FPGA 보드 설정과 회로 합성, 비트스트림 생성을 순서대로 수행할 수 있으며, 소스 코드(410)를 통해 설계된 논리 회로를 불러와 합성할 수 있다. 명령어(430)는 설계한 논리 회로의 배치 및 배선을 제어할 수 있으며, 명령어(430)를 기반으로 도 3의 CLB 타일 구조(310)에 도시된 바와 같이 x축 및 y축 좌표로 표현된 FPGA 타일 구조에 맞게 해당 타일에 설계한 논리 회로가 배치될 수 있다. 논리 회로의 입/출력 구조를 고정하면 논리 회로의

배치 변경시 그 구조를 유지한 채 위치만이 변경될 수 있다.

[0061] 도 5는 본 명세서의 일 실시 예에 따른 서로 다른 LUT에 배치된 논리 회로에 대응하는 비트스트림 및 비교표를 예시적으로 나타내는 도면이다. 도 5를 참조하면, 동일한 논리 회로를 서로 다른 CLB 타일의 서로 다른 LUT에 배치하여 생성된 두 비트스트림의 일부(510) 및 두 비트스트림의 데이터를 비교한 표(520)가 도시된다.

[0062] 일 실시 예에서, 논리 회로의 구성을 포함하는 비트스트림이 생성할 때, 비트스트림의 각 LUT에 대응하는 주소에 설계 데이터가 입력될 수 있다. 6비트의 입력 신호로 구성되는 LUT는 64개의 경우의 수를 가지며, 비트스트림 상에 대응하는 데이터의 크기는 총 8바이트이다. LUT의 설계 데이터는 비트스트림에 사용될 때 임의의 규칙을 가지고 변환되어 입력될 수 있다. 동일한 논리 회로를 두 개의 서로 다른 타일에 배치한 후 각각에 대한 비트스트림을 생성할 경우, 도 5의 표(520)에 도시된 바와 같이 두 개의 비트스트림은 서로 다른 주소에서 동일한 데이터 간격과 값을 가질 수 있다. 시작 주소는 다르나 동일한 데이터 간격과 값을 가질 수 있다. 이를 통해, FPGA 내 동일한 구성 회로는 동일한 비트스트림 변환 규칙을 가진다는 것을 확인할 수 있고, 이를 통해 단일 LUT에 대한 비트스트림 데이터 패턴을 기반으로 FPGA 내 모든 LUT에 구현되는 회로 구성을 파악할 수 있다. 이를 위해서는 동일한 논리 회로를 서로 다른 LUT에 배치하여 테스트 비트스트림 세트를 생성하고, 생성된 테스트 비트스트림 세트에 단일 LUT에 대한 비트스트림 데이터 변환 패턴을 적용하여 비트스트림을 분석해야 하며, 이를 위한 비트스트림 자동 생성 및 분석 알고리즘에 대해 이하 도 6을 통해 설명한다.

[0064] 도 6은 본 명세서의 일 실시 예에 따른 비트스트림 테스트 세트를 자동으로 생성 및 분석하는 방법의 흐름을 나타내는 순서도이다.

[0065] 도 6의 순서도를 참조하면, 전자 장치(100)는 FPGA 설계 소프트웨어 도구를 통해 FPGA 내 LUT 목록(601)을 획득하고, 다수의 LUT에 대응하는 논리식 설계 및 활성화 여부를 제어하는 소프트웨어 구동 명령어(602)를 확인할 수 있다.

[0066] 그 다음, 일 실시 예에서, 전자 장치(100)는 명령어를 목록에 적용하여 복수의 LUT에 대한 비트스트림 테스트 세트(604)를 생성할 수 있다. 전자 장치(100)는 PLP 역공학을 사용하여 FPGA 내 임의의 LUT(예를 들어, 제1 LUT)에 대한 비트스트림 상 데이터 패턴(예를 들어, 제1 분석 정보)을 획득한 후, 비트스트림 자동 생성 알고리즘(603)을 통해, FPGA 설계 소프트웨어 도구로부터 획득한 LUT 목록과 비트스트림 생성 명령어를 이용하여 동일한 논리 회로를 서로 다른 LUT에 배치한 비트스트림 테스트 세트(604)를 생성할 수 있다. 즉, 비트스트림 테스트 세트(604)는 제1 분석 정보에 대응하는 논리 회로를 FPGA 내 복수의 LUT 중 제1 LUT와 상이한 LUT에 배치하도록 생성될 수 있다. 전자 장치(100)는 이와 같은 테스트 비트스트림을 목록(601)에 포함된 모든 LUT에 대해 생성할 수 있다.

[0067] 일 실시 예에서, 전자 장치(100)는 제1 분석 정보를 비트스트림 테스트 세트(604)에 적용하여 복수의 LUT에 대한 제2 분석 정보를 획득할 수 있다. 제2 분석 정보는 복수의 LUT의 비트스트림 상 주소 정보를 포함할 수 있다. 모든 LUT에 대한 비트스트림 테스트 세트가 생성되면 전자 장치(100)는 기존 PLP 역공학을 통해 생성된 단일 LUT 비트스트림 데이터 패턴(605)을 각 테스트 비트스트림 내에서 자동으로 탐색하는 비트스트림 자동 분석 알고리즘(606)을 작성할 수 있다. 비트스트림 자동 분석 알고리즘(606)을 통해, 전자 장치(100)는 각 비트스트림별, 즉 각 LUT별 비트스트림상 시작 주소 정보(607)를 최종적으로 획득할 수 있다. 전자 장치(100)는 획득된 시작 주소 정보를 데이터베이스 내에 정리하고 이를 실제 비트스트림을 대상으로 하는 역공학에 활용할 수 있다.

[0069] 도 7은 본 명세서의 일 실시 예에 따른 자동화 기법을 통한 FPGA 내 논리 회로 구성의 역공학 과정을 예시적으로 나타내는 도면(700)이다.

[0070] 일 실시 예에서, 전자 장치(100)는 PLP 역공학을 통해 획득한 단일 LUT에 대한 비트스트림 변환 규칙 및 역공학 자동화 기법을 통해 획득한 FPGA LUT 주소 데이터베이스를 활용하여 도 6에 도시된 바와 같은 완전한 FPGA 논리 회로의 역공학을 수행할 수 있다. 일 실시 예에서, 전자 장치(100)는 복수의 LUT의 비트스트림 상 주소 정보를 기반으로 비트스트림을 분리하고, 제1 분석 정보에 포함된 단일 LUT에 대한 비트스트림 변환 규칙을 기반으로 복수의 LUT 각각에 구현된 논리 회로를 획득할 수 있다. 전자 장치(100)는 실제 비트스트림이 주어질 때, 데이

터베이스에 정리된 LUT 별 주소를 확인하여 각 LUT에 대응하는 데이터를 분리할 수 있다. 다음으로, PLP 역공학을 통해 획득한 비트스트림 변환 패턴을 적용하여, 각 LUT에 구현된 논리 회로를 획득할 수 있다. 이와 같이 획득한 모든 LUT 논리 회로를 종합하면 전자 장치(100)는 시스템 전체의 회로 구성을 파악할 수 있게 될 것이다.

[0071] 도 7을 참조하면, 전자 장치(100)는 역공학을 수행하고자 하는 FPGA의 실제 비트스트림(710)을 추출하고, 비트스트림 테스트 세트와 자동 분석 알고리즘을 통해 획득한 FPGA의 LUT 주소 데이터베이스(720)를 기반으로 비트스트림(710)의 데이터를 LUT 주소 별로 분리함으로써 분리된 LUT 데이터(730)를 획득할 수 있다. 전자 장치(100)는 분리된 LUT 데이터에 PLP 역공학을 통해 획득한 단일 LUT의 비트스트림 변환 규칙을 적용하여 각 LUT에 구현된 논리 회로(740)를 획득할 수 있으며, 각 LUT 구성 논리 회로(740)를 종합하여 전체 시스템 논리 회로(750)을 획득할 수 있다.

[0073] 도 8은 본 명세서의 일 실시 예에 따른 전자 장치가 FPGA를 분석하는 방법의 흐름을 나타내는 순서도이다. 도 8에 도시된 각 단계의 수행 주체는 도 1에 도시된 전자 장치(100)를 포함할 수 있으며, 전술한 내용과 중복되는 설명이 이하에서는 생략될 수 있다.

[0074] 단계 S810에서, 전자 장치는 FPGA의 비트스트림으로부터, FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보를 획득할 수 있다. 일 실시 예에서, 제1 분석 정보는 제1 LUT에 대한 비트스트림 변환 규칙 정보 및 제1 LUT의 비트스트림 상 주소 정보를 포함할 수 있다. 전자 장치는 FPGA에 포함된 단일 LUT에 대한 역공학 방법을 수행함으로써 단일 LUT에 대한 비트스트림 변환 규칙과 데이터 주소 정보를 포함하는 분석 정보를 획득할 수 있다.

[0075] 단계 S820에서, 전자 장치는 FPGA 설계 소프트웨어를 사용하여 FPGA에 포함된 복수의 LUT의 목록을 획득할 수 있다. 일 실시 예에서, 복수의 LUT의 목록은 FPGA에 포함된 CLB 타일 구조에 대한 정보, 및 CLB 타일 내 포함된 LUT에 대한 정보를 포함할 수 있다.

[0076] 단계 S830에서, 전자 장치는 FPGA 설계 소프트웨어를 FPGA 상에 실행하기 위한 명령어를 확인할 수 있다. 소프트웨어 구동 명령어를 사용하면 설계 회로의 세부 배치와 연결을 임의로 설정할 수 있으며, 파일 생성부터 비트스트림 생성까지의 동작을 자동으로 수행할 수 있다. 일 실시 예에서, 명령어는 FPGA 보드 설정 및 비트스트림 생성을 위한 제1 명령어와, 회로 배치 및 배선 설정을 위한 제2 명령어 중 적어도 하나를 포함할 수 있다. 일 실시 예에서, 제1 명령어는 사용하고자 하는 FPGA 보드 설정과 회로 합성, 비트스트림 생성을 순서대로 수행할 수 있으며, 소스 코드를 통해 설계된 논리 회로를 불러와 합성할 수 있다. 일 실시 예에서, 제2 명령어는 설계한 논리 회로의 배치 및 배선을 제어할 수 있으며, 제2 명령어를 기반으로 예를 들어, 도 3의 CLB 타일 구조(310)에 도시된 바와 같이 x축 및 y축 좌표로 표현된 FPGA 타일 구조에 맞게 해당 타일에 설계한 논리 회로가 배치될 수 있다. 논리 회로의 입/출력 구조를 고정하면 논리 회로의 배치 변경시 그 구조를 유지한 채 위치만이 변경될 수 있다.

[0077] 단계 S840에서, 전자 장치는 명령어를 목록에 적용하여 복수의 LUT에 대한 비트스트림 테스트 세트를 생성할 수 있다. 전자 장치는 PLP 역공학을 사용하여 FPGA 내 임의의 제1 LUT에 대한 제1 분석 정보를 획득한 후, 비트스트림 자동 생성 알고리즘을 통해, FPGA 설계 소프트웨어 도구로부터 획득한 LUT 목록과 비트스트림 생성 명령어를 이용하여 동일한 논리 회로를 서로 다른 LUT에 배치한 비트스트림 테스트 세트를 생성할 수 있다. 즉, 비트스트림 테스트 세트는 제1 분석 정보에 대응하는 논리 회로를 FPGA 내 복수의 LUT 중 제1 LUT와 상이한 LUT에 배치하도록 생성될 수 있다. 전자 장치는 이와 같은 테스트 비트스트림을 목록에 포함된 복수의 LUT 모두에 대해 생성할 수 있다.

[0078] 단계 S850에서, 전자 장치는 제1 분석 정보를 비트스트림 테스트 세트에 적용하여 복수의 LUT에 대한 제2 분석 정보를 획득할 수 있다. 제2 분석 정보는 복수의 LUT의 비트스트림 상 주소 정보를 포함할 수 있다. 모든 LUT에 대한 비트스트림 테스트 세트가 생성되면 전자 장치는 기존 PLP 역공학을 통해 생성된 단일 LUT 비트스트림 데이터 패턴을 각 테스트 비트스트림 내에서 자동으로 탐색하는 비트스트림 자동 분석 알고리즘을 작성할 수 있다. 비트스트림 자동 분석 알고리즘을 통해, 전자 장치는 각 비트스트림별, 즉 각 LUT별 비트스트림상 시작 주소 정보를 최종적으로 획득할 수 있다. 전자 장치는 획득된 시작 주소 정보를 데이터베이스 내에 정리하고 이를 실제 비트스트림을 대상으로 하는 역공학에 활용할 수 있다.

[0080] 전술한 실시 예들에 따른 장치는 프로세서, 프로그램 데이터를 저장하고 실행하는 메모리, 디스크 드라이브와

같은 영구 저장부(permanent storage), 외부 장치와 통신하는 통신 포트, 터치 패널, 키(key), 버튼 등과 같은 사용자 인터페이스 장치 등을 포함할 수 있다. 소프트웨어 모듈 또는 알고리즘으로 구현되는 방법들은 상기 프로세서상에서 실행 가능한 컴퓨터가 읽을 수 있는 코드들 또는 프로그램 명령들로서 컴퓨터가 읽을 수 있는 기록 매체 상에 저장될 수 있다. 여기서 컴퓨터가 읽을 수 있는 기록 매체로 마그네틱 저장 매체(예컨대, ROM(read-only memory), RAM(random-Access memory), 플로피 디스크, 하드 디스크 등) 및 광학적 판독 매체(예컨대, 시디롬(CD-ROM), 디브이디(DVD: Digital Versatile Disc)) 등이 있다. 컴퓨터가 읽을 수 있는 기록 매체는 네트워크로 연결된 컴퓨터 시스템들에 분산되어, 분산 방식으로 컴퓨터가 판독 가능한 코드가 저장되고 실행될 수 있다. 매체는 컴퓨터에 의해 판독가능하며, 메모리에 저장되고, 프로세서에서 실행될 수 있다.

[0081] 본 실시 예는 기능적인 블록 구성들 및 다양한 처리 단계들로 나타내어질 수 있다. 이러한 기능 블록들은 특정 기능들을 실행하는 다양한 개수의 하드웨어 또는/및 소프트웨어 구성들로 구현될 수 있다. 예를 들어, 실시 예는 하나 이상의 마이크로프로세서들의 제어 또는 다른 제어 장치들에 의해서 다양한 기능들을 실행할 수 있는, 메모리, 프로세싱, 로직(logic), 룩 업 테이블(look-up table) 등과 같은 직접 회로 구성들을 채용할 수 있다. 구성 요소들이 소프트웨어 프로그래밍 또는 소프트웨어 요소들로 실행될 수 있는 것과 유사하게, 본 실시 예는 데이터 구조, 프로세스들, 루틴들 또는 다른 프로그래밍 구성들의 조합으로 구현되는 다양한 알고리즘을 포함하여, C, C++, 자바(Java), 어셈블러(assembler) 등과 같은 프로그래밍 또는 스크립팅 언어로 구현될 수 있다. 기능적인 측면들은 하나 이상의 프로세서들에서 실행되는 알고리즘으로 구현될 수 있다. 또한, 본 실시 예는 전자적인 환경 설정, 신호 처리, 및/또는 데이터 처리 등을 위하여 종래 기술을 채용할 수 있다. “매커니즘”, “요소”, “수단”, “구성”과 같은 용어는 넓게 사용될 수 있으며, 기계적이고 물리적인 구성들로서 한정되는 것은 아니다. 상기 용어는 프로세서 등과 연계하여 소프트웨어의 일련의 처리들(routines)의 의미를 포함할 수 있다.

[0082] 한편, 본 명세서와 도면에는 본 발명의 바람직한 실시 예에 대하여 개시하였으며, 비록 특정 용어들이 사용되었으나, 이는 단지 본 발명의 기술 내용을 쉽게 설명하고 발명의 이해를 돕기 위한 일반적인 의미에서 사용된 것이지, 본 발명의 범위를 한정하고자 하는 것은 아니다. 여기에 개시된 실시 예 외에도 본 발명의 기술적 사상에 바탕을 둔 다른 변형 예들이 실시 가능하다는 것은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에게 자명한 것이다.

부호의 설명

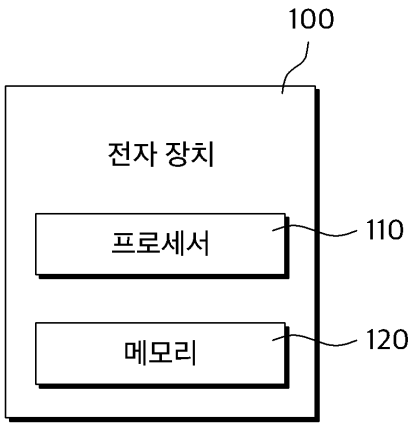
[0083] 100: 전자 장치

110: 프로세서

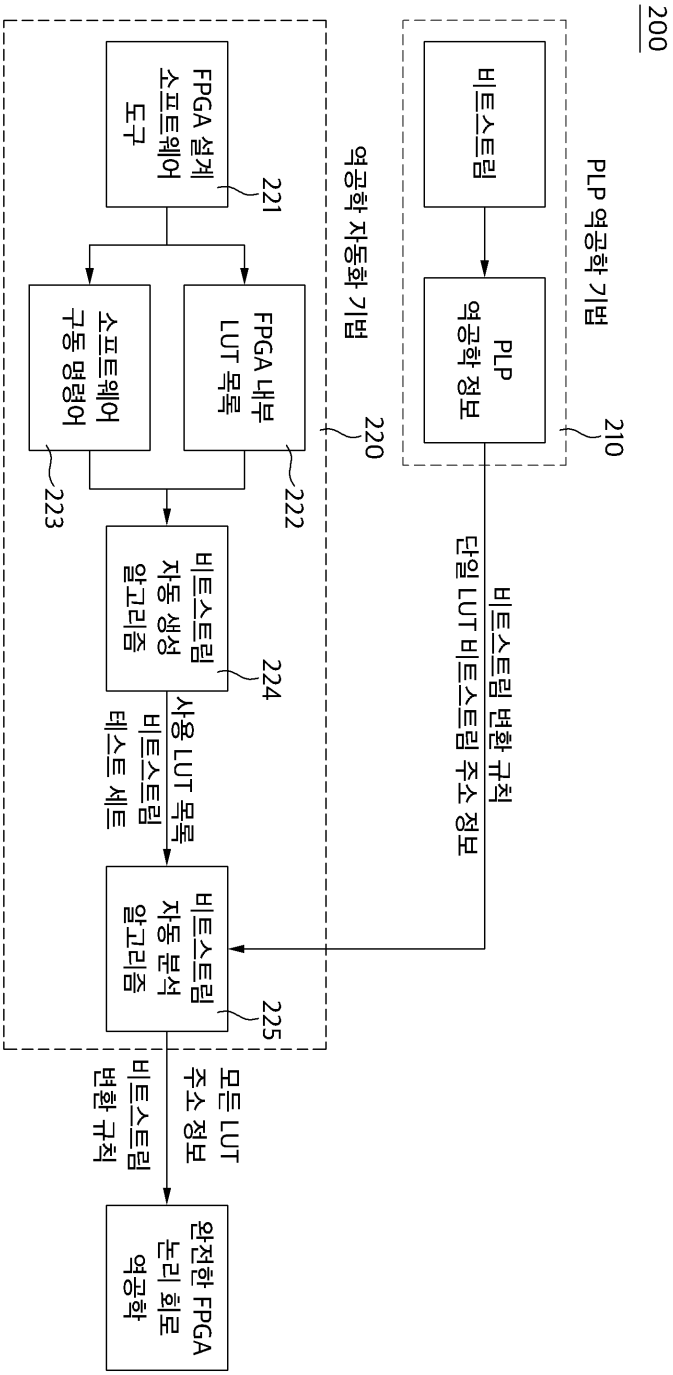
120: 메모리

도면

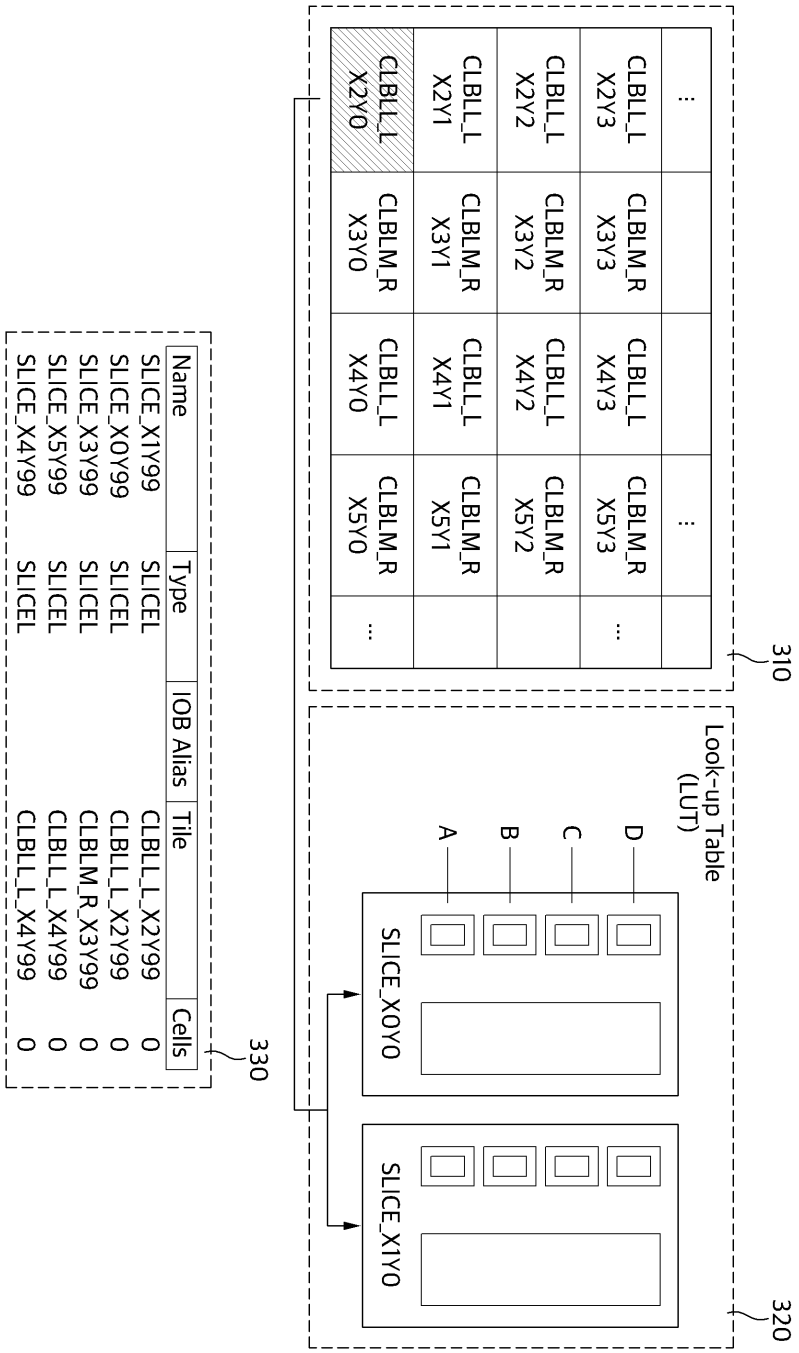
도면1



도면2


200
PLP 역공학 기법
210
비트스트림
PLP 역공학 정보
단일 LUT 비트스트림 주소 정보
비트스트림 변환 규칙
역공학 자동화 기법
220
FPGA 설계 소프트웨어 도구
221
FPGA 내부 LUT 목록
222
소프트웨어 구동 명령어
223
비트스트림 자동 생성 알고리즘
224
사용 LUT 목록
비트스트림 테스트 세트
비트스트림 자동 분석 알고리즘
225
모든 LUT 주소 정보
비트스트림 변환 규칙
완전한 FPGA 논리 회로 역공학

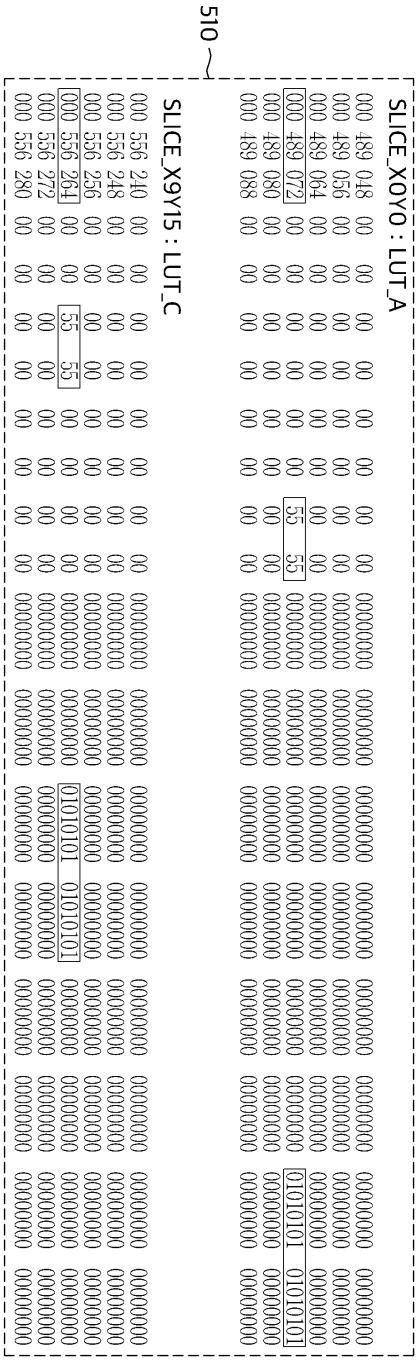
도면3


310
CLBLL_L X2Y3
CLBLM_R X3Y3
CLBLL_L X4Y3
CLBLM_R X5Y3
CLBLL_L X2Y2
CLBLM_R X3Y2
CLBLL_L X4Y2
CLBLM_R X5Y2
CLBLL_L X2Y1
CLBLM_R X3Y1
CLBLL_L X4Y1
CLBLM_R X5Y1
CLBLL_L X2Y0
CLBLM_R X3Y0
CLBLL_L X4Y0
CLBLM_R X5Y0
320
Look-up Table (LUT)
A
B
C
D
SLICE_X0Y0
SLICE_X1Y0
330
Name
Type
IOB Alias
Tile
Cells
SLICE_X1Y99
SLICEL
CLBLL_L_X2Y99
0
SLICE_X0Y99
SLICEL
CLBLL_L_X2Y99
0
SLICE_X3Y99
SLICEL
CLBLM_R_X3Y99
0
SLICE_X5Y99
SLICEL
CLBLL_L_X4Y99
0
SLICE_X4Y99
SLICEL
CLBLL_L_X4Y99
0

도면4

410
논리 회로 소스 코드 예시
module Project_Source (
input a, b,
output c
);
assign c = (a ^ b);
endmodule
a
b
c
420
FPGA 보드 설정 및 비트스트림 생성 명령어
create_project Project_Name ./ -part xc7a12tcpq238-l
add_files -fileset sources_1 -norecurse ./Project_Source.v
add_files -fileset constrs_1 -norecurse ./Project_Const.xdc
set_property STEPS.WRITE_BITSTREAM.ARGS.BIN_FILE true [get_runs impl_1]
launch_runs impl_1 -to_step write_bitstream -jobs 6
close_project
430
회로 배치 및 배선 설정 명령어
set_property BEL A6LUT [get_cells c_OBUF_inst_i_1]
set_property LOC SLICE_X0Y0 [get_cells c_OBUF_inst_i_1]
set_property PACKAGE_PIN U14 [get_ports a]
set_property PACKAGE_PIN W16 [get_ports b]
set_property PACKAGE_PIN W15 [get_ports c]
set_property IOSTANDARD LVCMOS18 [get_ports a]
set_property IOSTANDARD LVCMOS18 [get_ports b]
set_property IOSTANDARD LVCMOS18 [get_ports c]
set_property LOCK_PINS {I0:A1 I1:A2} [get_cells c_OBUF_inst_i_1]

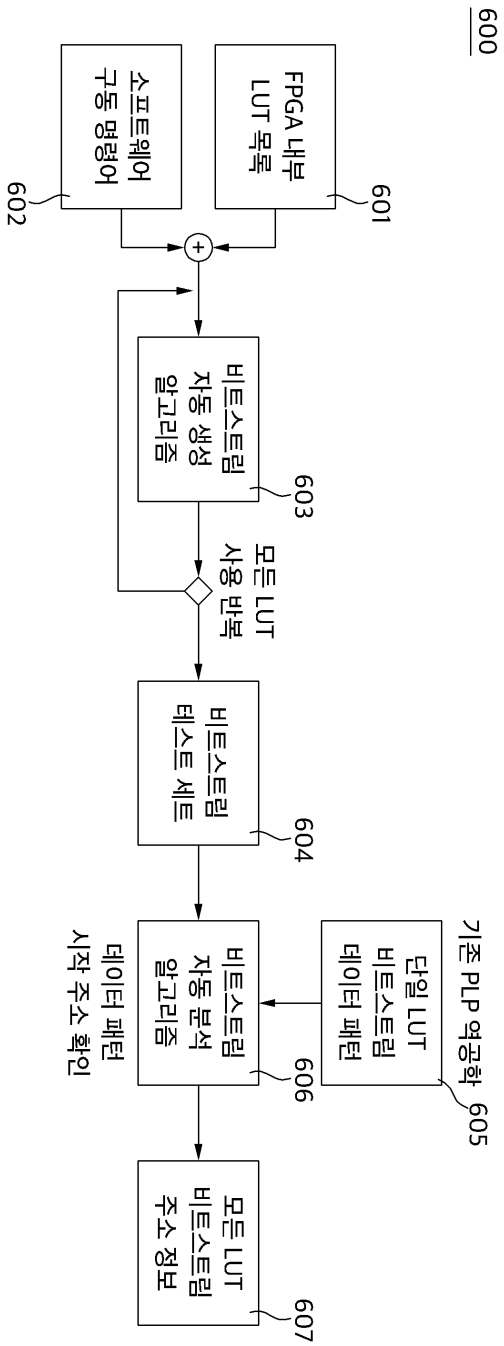
도면5



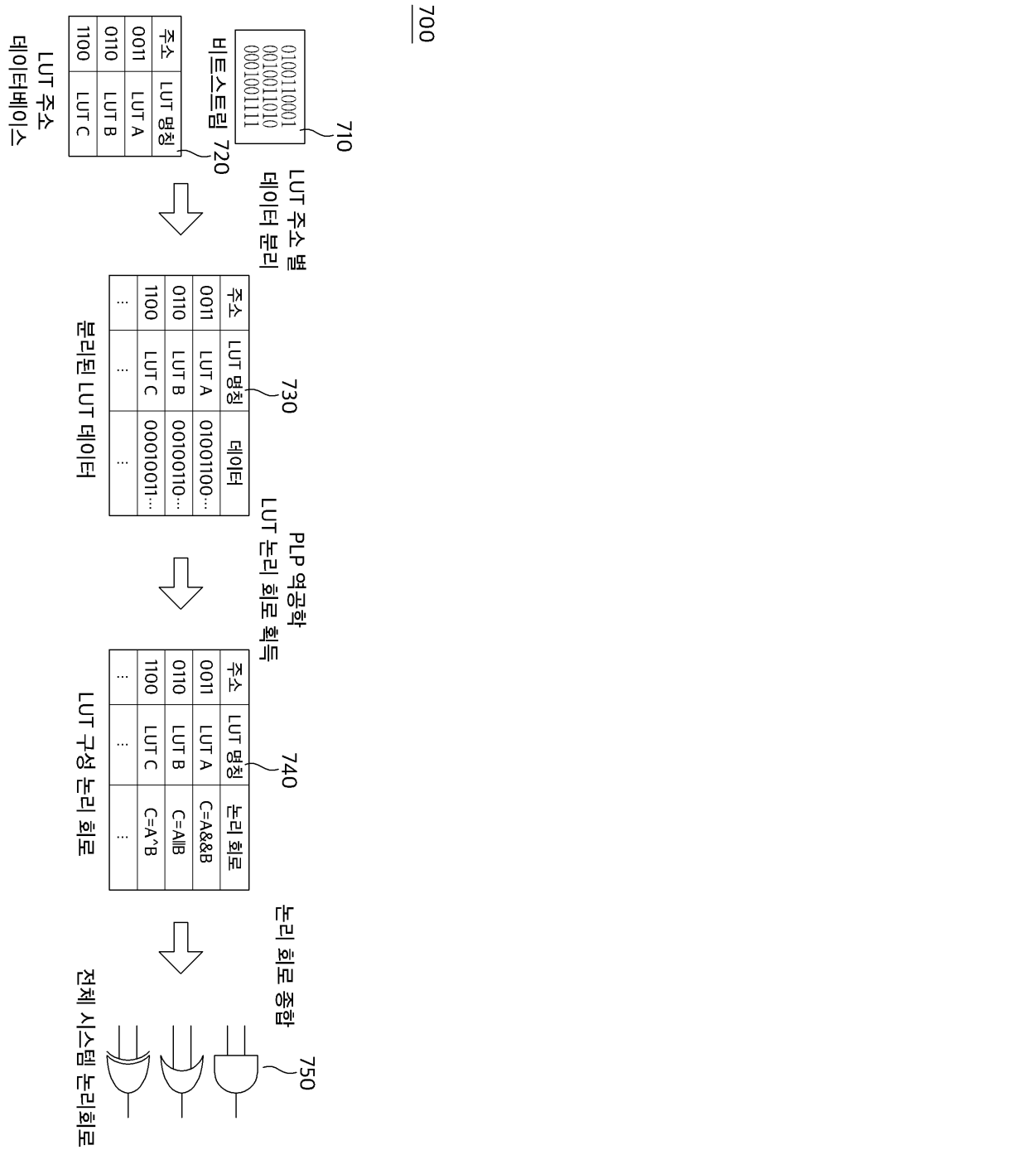
520

타일 슬라이스	LUT	비트스트림 주소	비트스트림 데이터
SLICE_X0Y0	LUT_A	489,078	55 55
		489,482	AA AA
		489,886	AA AA
		490,290	55 55
SLICE_X9Y15	LUT_C	556,266	55 55
		556,670	AA AA
		557,074	AA AA
		557,478	55 55

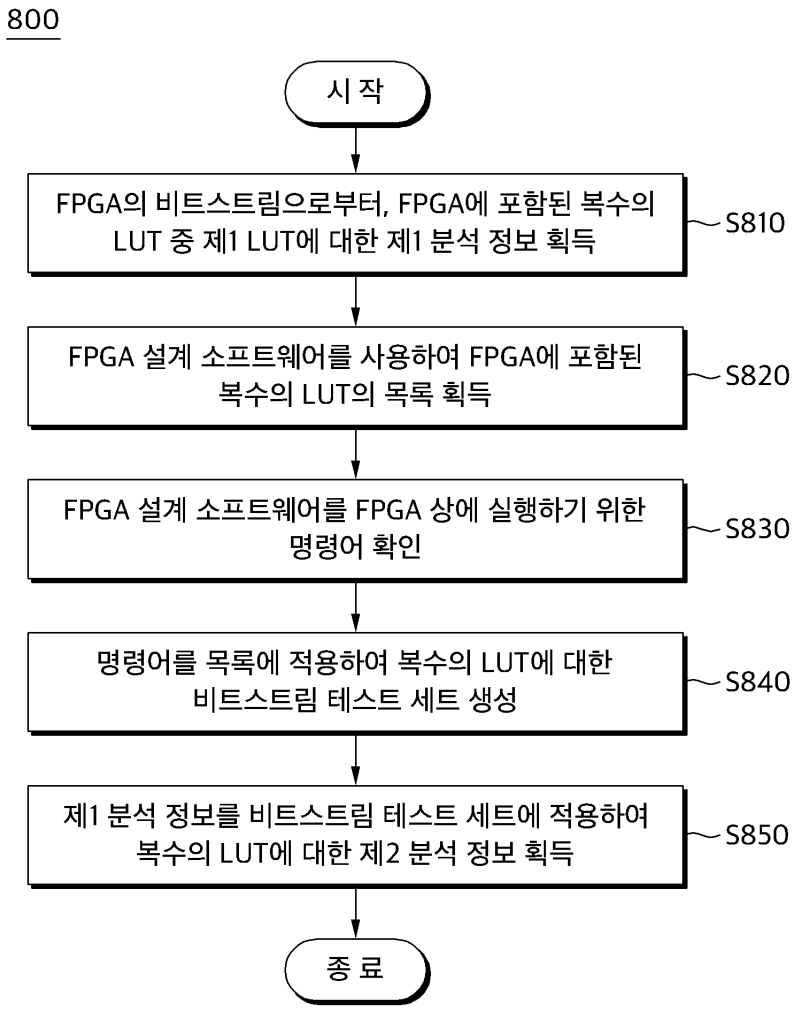
도면6


600
FPGA 내부
LUT 목록
601
소프트웨어
구동 명령어
602
비트스트림
자동 생성
알고리즘
603
모든 LUT
사용 반복
비트스트림
테스트 세트
604
기존 PLP 역공학
단일 LUT
비트스트림
데이터 패턴
605
비트스트림
자동 분석
알고리즘
606
데이터 패턴
시작 주소 확인
모든 LUT
비트스트림
주소 정보
607

도면7


700
710
0100110001
0010011010
0001001111
비트스트림 720
주소 | LUT 명칭
0011 | LUT A
0110 | LUT B
1100 | LUT C
LUT 주소
데이터베이스
LUT 주소 별
데이터 분리
730
주소 | LUT 명칭 | 데이터
0011 | LUT A | 01001100…
0110 | LUT B | 00100110…
1100 | LUT C | 00010011…
분리된 LUT 데이터
PLP 역공학
LUT 논리 회로 획득
740
주소 | LUT 명칭 | 논리 회로
0011 | LUT A | C=A&&B
0110 | LUT B | C=A|B
1100 | LUT C | C=A^B
LUT 구성 논리 회로
논리 회로 종합
750
전체 시스템 논리회로

도면8


800
시 작
FPGA의 비트스트림으로부터, FPGA에 포함된 복수의 LUT 중 제1 LUT에 대한 제1 분석 정보 획득
S810
FPGA 설계 소프트웨어를 사용하여 FPGA에 포함된 복수의 LUT의 목록 획득
S820
FPGA 설계 소프트웨어를 FPGA 상에 실행하기 위한 명령어 확인
S830
명령어를 목록에 적용하여 복수의 LUT에 대한 비트스트림 테스트 세트 생성
S840
제1 분석 정보를 비트스트림 테스트 세트에 적용하여 복수의 LUT에 대한 제2 분석 정보 획득
S850
종 료