(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2023-0026726
(43) 공개일자 2023년02월27일

(51) 국제특허분류(Int. Cl.)
G06F 11/30 (2006.01) *G06F 15/78* (2006.01)
G06F 9/30 (2018.01) *G06F 9/38* (2006.01)
(52) CPC특허분류
G06F 11/3079 (2013.01)
G06F 11/3037 (2013.01)
(21) 출원번호 10-2021-0108693
(22) 출원일자 2021년08월18일
심사청구일자 2021년08월18일

(71) 출원인
세종대학교산학협력단
서울특별시 광진구 능동로 209 (군자동, 세종대학교)
(72) 발명자
박기웅
서울특별시 광진구 능동로17길 21, 304호(화양동)
정혜림
서울특별시 동작구 사당로16사길 47, 102호(사당동)
최기철
서울특별시 광진구 동일로28길 49-9, 703호(화양동)
(74) 대리인
양성보

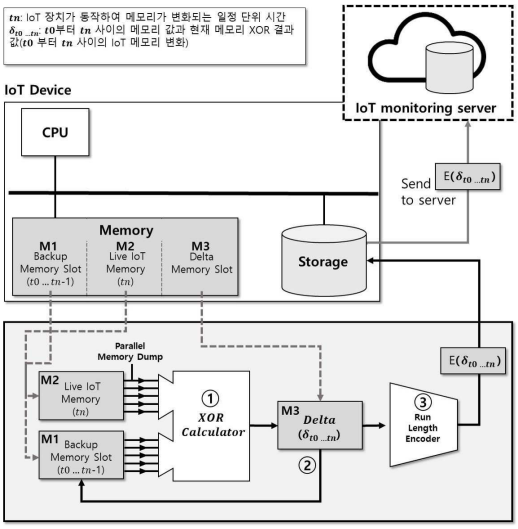
전체 청구항 수 : 총 15 항

(54) 발명의 명칭 **병렬 회로를 이용한 저지연 방식의 실시간 메모리 연속 수집 기법 및 IoT 메모리 재현 기술**

(57) 요 약

병렬회로를 이용한 저지연 방식의 실시간 메모리 연속 수집 기법 및 IoT 메모리 재현 기술이 개시된다. 일 실시예에 따른 컴퓨터 시스템에 의해 수행되는 메모리 연속 수집 방법은, 구동 중인 IoT 장치를 대상으로 상기 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집하는 단계; 및 상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득하는 단계를 포함할 수 있다.

대 표 도 - 도3



(52) CPC특허분류
G06F 15/7867 (2013.01)
G06F 9/30029 (2013.01)
G06F 9/30043 (2013.01)
G06F 9/3885 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711126051
과제번호 2019-0-00426-003
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 정보보호핵심원천기술개발(R&D,정보화)
연구과제명 IoT 기반 이식-침습형 고위험 의료장치를 위한 능동형 킬 스위치 및 바이오 마커 활용 방어 시스템 개발
기 여 율 1/1
과제수행기관명 국민대학교산학협력단
연구기간 2021.01.01 ~ 2021.12.31

명 세 서

청구범위

청구항 1

컴퓨터 시스템에 의해 수행되는 메모리 연속 수집 방법에 있어서,

구동 중인 IoT 장치를 대상으로 상기 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집하는 단계; 및

상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 2

제1항에 있어서,

상기 수집하는 단계는,

상기 IoT 장치에 구성된 메모리에 메모리 덤프 연산을 위한 복수 개의 슬롯을 생성하고, 상기 생성된 복수 개의 슬롯을 이용하여 상기 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프를 수행하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 3

제1항에 있어서,

상기 수집하는 단계는,

제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 메모리 데이터를 0으로 초기화하고, 제1 메모리 덤프를 수행할 경우, FPGA 기반의 병렬 회로를 통해 제2 메모리 슬롯에 저장된 메모리 데이터를 상기 0으로 초기화된 제1 메모리 슬롯으로 전달하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 4

제3항에 있어서,

상기 수집하는 단계는,

상기 제1 메모리 덤프를 수행한 이후, 제N (N은 2이상의 자연수) 메모리 덤프를 수행할 때, 제N (N은 2이상의 자연수) 메모리 덤프 과정에서의 상기 제2 메모리 슬롯의 메모리 데이터와 상기 N-1(N은 2이상의 자연수) 메모리 덤프 과정에서 기록된 상기 제1 메모리 슬롯의 메모리 덤프 데이터를 XOR 연산한 결과 데이터를 획득하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 5

제4항에 있어서,

상기 수집하는 단계는,

상기 획득된 결과 데이터를 제3 메모리 슬롯에 저장하고, 상기 제3 메모리 슬롯에 저장된 결과 데이터가 상기 제1 메모리 슬롯에 저장되고, 상기 제3 메모리 슬롯에 저장된 결과 데이터에 RLE(Run length Encoder) 압축 알고리즘을 사용하여 압축하고, 상기 압축된 결과 데이터를 스토리지에 저장하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 6

제1항에 있어서,

상기 수집하는 단계는,

상기 수행된 메모리 덤프를 통해 현재 시점의 메모리 덤프 데이터와 현재 시점을 기준으로 이전 시점의 메모리 덤프 데이터의 비교를 통해 변화 데이터를 수집하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 7

제1항에 있어서,

상기 획득하는 단계는,

상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 상기 IoT 장치의 분석이 요청된 특정 시점을 기준으로 이전 시점의 변화 데이터를 추출하고, 상기 추출된 특정 시점의 변화 데이터와 상기 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 원본 메모리 데이터를 획득하는 단계

를 포함하는 메모리 연속 수집 방법.

청구항 8

제1항 내지 제7항 중 어느 한 항의 메모리 연속 수집 방법을 상기 컴퓨터 시스템에 실행시키기 위해 비-일시적인 컴퓨터 판독가능한 기록 매체에 저장되는 컴퓨터 프로그램.

청구항 9

컴퓨터 시스템에 있어서,

메모리에 포함된 컴퓨터 판독가능한 명령들을 실행하도록 구성된 적어도 하나의 프로세서

를 포함하고,

상기 적어도 하나의 프로세서는,

구동 중인 IoT 장치를 대상으로 상기 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집하고,

상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 10

제9항에 있어서,

상기 적어도 하나의 프로세서는,

상기 IoT 장치에 구성된 메모리에 메모리 덤프 연산을 위한 복수 개의 슬롯을 생성하고, 상기 생성된 복수 개의 슬롯을 이용하여 상기 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프를 수행하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 11

제9항에 있어서,

상기 적어도 하나의 프로세서는,

제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 메모리 데이터를 0으로 초기화하고, 제1 메모리 덤프를 수행할 경우, FPGA 기반의 병렬 회로를 통해 제2 메모리 슬롯에 저장된 메모리 데이터를 상기 0으로 초기화된 제1 메모리 슬롯으로 전달하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 12

제11항에 있어서,

상기 적어도 하나의 프로세서는,

상기 제1 메모리 덤프를 수행한 이후, 제N (N은 2이상의 자연수) 메모리 덤프를 수행할 때, 제N (N은 2이상의 자연수) 메모리 덤프 과정에서의 상기 제2 메모리 슬롯의 메모리 데이터와 상기 N-1(N은 2이상의 자연수) 메모리 덤프 과정에서 기록된 상기 제1 메모리 슬롯의 메모리 덤프 데이터를 XOR 연산한 결과 데이터를 획득하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 13

제12항에 있어서,

상기 적어도 하나의 프로세서는,

상기 획득된 결과 데이터를 제3 메모리 슬롯에 저장하고, 상기 제3 메모리 슬롯에 저장된 결과 데이터가 상기 제1 메모리 슬롯에 저장되고, 상기 제3 메모리 슬롯에 저장된 결과 데이터에 RLE(Run length Encoder) 압축 알고리즘을 사용하여 압축하고, 상기 압축된 결과 데이터를 스토리지에 저장하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 14

제9항에 있어서,

상기 적어도 하나의 프로세서는,

상기 수행된 메모리 덤프를 통해 현재 시점의 메모리 덤프 데이터와 현재 시점을 기준으로 이전 시점의 메모리 덤프 데이터의 비교를 통해 변화 데이터를 수집하는

것을 특징으로 하는 컴퓨터 시스템.

청구항 15

제9항에 있어서,

상기 적어도 하나의 프로세서는,

상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 상기 IoT 장치의 분석이 요청된 특정 시점을 기준으로 이전 시점의 변화 데이터를 추출하고, 상기 추출된 특정 시점의 변화 데이터와 상기 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 원본 메모리 데이터를 획득하는

것을 특징으로 하는 컴퓨터 시스템.

발명의 설명

기 술 분 야

[0001] 아래의 설명은 IoT 장치의 이상징후 탐지를 위한 메모리 수집 및 분석 기술에 관한 것이다.

배 경 기 술

[0003] 최근 스마트 시티와 스마트 빌딩 등 IoT의 활용 용도가 늘어남에 따라 다수의 IoT 장치가 곳곳에 적용되어 생활에 편리함을 제공하고 있다. 다양한 위치에 존재하는 다수의 IoT 장치를 관리하기 위해 스마트 관제 센터를 설립하여 IoT 장치의 문제 발생 여부와 사이버 공격을 모니터링하고 있지만, 원격에 존재하는 낮은 성능의 IoT 장치 특성상 에이전트 설치의 어려움으로 점차 고도화되는 사이버 공격, 기기 고장 등 IoT 장치 내부의 문제 발생 시 탐지의 어려움이 존재한다. 또한, 메모리 중심 컴퓨팅 기술의 발전으로 다수의 메모리 노드를 공유하여 사용하는 방식의 체제가 연구되고 있기는 하나 효율적인 메모리 실시간 모니터링이 필요한 상황이다. 따라서 원격에서 동작 중인 IoT 장치의 이상징후 탐지 및 원인을 찾기 위한 심층 분석 및 실시간 모니터링을 위해서는 동작 중인 IoT 장치의 연속적인 메모리 수집 및 분석 기술이 요구된다. 하지만 메모리 덤프 연산은 서스펜스 현상(시스템 전체 일시 정지 현상)을 유발하며 덤프해야 하는 메모리 용량에 따라 서스펜스 현상 시간을 증가시켜 정상적인 IoT 장치의 동작을 방해한다. 또한, 수집된 시계열 메모리 덤프 데이터를 원격으로 보내기 위해서는 메모리 덤프 데이터를 임시로 저장할 공간이 필요하여 동작 중인 IoT 장치의 사용 가능 저장 공간이 부족해지는 문제가 존재한다.

발명의 내용

해결하려는 과제

[0005] IoT 장치를 심층적으로 분석할 때 메모리 덤프 연산으로 인해 메모리 덤프 주기마다 발생하는 서스펜스 현상 시간을 최소화하기 위하여 메모리의 병렬 동작을 통한 저지연 방식으로 메모리 데이터를 수집하는 방법 및 시스템을 제공할 수 있다.

[0006] 현 시점의 메모리 덤프 데이터와 이전 주기의 메모리 덤프 데이터를 비교해 변화된 부분만 수집함으로써 경량화된 메모리 덤프 데이터를 확보하고, 변화된 부분에 대한 정보를 가지고 있는 델타 측정값을 이용하여 특정 시점의 메모리 덤프 데이터를 역으로 획득해 동작 중인 IoT 장치에 미치는 영향을 최소화하는 이상징후 탐지 위한 방법 및 시스템을 제공할 수 있다.

과제의 해결 수단

[0008] 컴퓨터 시스템에 의해 수행되는 메모리 연속 수집 방법은, 구동 중인 IoT 장치를 대상으로 상기 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집하는 단계; 및 상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득하는 단계를 포함할 수 있다.

[0009] 상기 수집하는 단계는, 상기 IoT 장치에 구성된 메모리에 메모리 덤프 연산을 위한 복수 개의 슬롯을 생성하고, 상기 생성된 복수 개의 슬롯을 이용하여 상기 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프를 수행하는 단계를 포함할 수 있다.

[0010] 상기 수집하는 단계는, 제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 메모리 데이터를 0으로 초기화하고, 제1 메모리 덤프를 수행할 경우, FPGA 기반의 병렬 회로를 통해 제2 메모리 슬롯에 저장된 메모리 데이터를 상기 0으로 초기화된 제1 메모리 슬롯으로 전달하는 단계를 포함할 수 있다.

[0011] 상기 수집하는 단계는, 상기 제1 메모리 덤프를 수행한 이후, 제N (N은 2이상의 자연수) 메모리 덤프를 수행할 때, 제N (N은 2이상의 자연수) 메모리 덤프 과정에서의 상기 제2 메모리 슬롯의 메모리 데이터와 상기 N-1(N은 2이상의 자연수) 메모리 덤프 과정에서 기록된 상기 제1 메모리 슬롯의 메모리 덤프 데이터를 XOR 연산한 결과 데이터를 획득하는 단계를 포함할 수 있다.

[0012] 상기 수집하는 단계는, 상기 획득된 결과 데이터를 제3 메모리 슬롯에 저장하고, 상기 제3 메모리 슬롯에 저장된 결과 데이터가 상기 제1 메모리 슬롯에 저장되고, 상기 제3 메모리 슬롯에 저장된 결과 데이터에 RLE(Run length Encoder) 압축 알고리즘을 사용하여 압축하고, 상기 압축된 결과 데이터를 스토리지에 저장하는 단계를 포함할 수 있다.

[0013] 상기 수집하는 단계는, 상기 수행된 메모리 덤프를 통해 현재 시점의 메모리 덤프 데이터와 현재 시점을 기준으로 이전 시점의 메모리 덤프 데이터의 비교를 통해 변화 데이터를 수집하는 단계를 포함할 수 있다.

[0014] 상기 획득하는 단계는, 상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 상기 IoT 장치의 분석이 요청된 특정 시점을 기준으로 이전 시점의 변화 데이터를 추출하고, 상기 추출된 특정 시점의 변화 데이터와 상기 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 원본 메모리 데이터를 획득하는 단계를 포함할 수 있다.

[0015] 메모리 연속 수집 방법을 상기 컴퓨터 시스템에 실행시키기 위해 비-일시적인 컴퓨터 판독가능한 기록 매체에 저장되는 컴퓨터 프로그램을 포함할 수 있다.

[0016] 컴퓨터 시스템에 있어서, 메모리에 포함된 컴퓨터 판독가능한 명령들을 실행하도록 구성된 적어도 하나의 프로세서를 포함하고, 상기 적어도 하나의 프로세서는, 구동 중인 IoT 장치를 대상으로 상기 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집하고, 상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득할 수 있다.

[0017] 상기 적어도 하나의 프로세서는, 상기 IoT 장치에 구성된 메모리에 메모리 덤프 연산을 위한 복수 개의 슬롯을 생성하고, 상기 생성된 복수 개의 슬롯을 이용하여 상기 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프를 수행할 수 있다.

[0018] 상기 적어도 하나의 프로세서는, 제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 메모리 데이터를 0으로 초기화하고, 제1 메모리 덤프를 수행할 경우, FPGA 기반의 병렬 회로를 통해 제2 메모리 슬롯에 저장된 메모리 데이터를 상기 0으로 초기화된 제1 메모리 슬롯으로 전달할 수 있다.

[0019] 상기 적어도 하나의 프로세서는, 상기 제1 메모리 덤프를 수행한 이후, 제N (N은 2이상의 자연수) 메모리 덤프를 수행할 때, 제N (N은 2이상의 자연수) 메모리 덤프 과정에서의 상기 제2 메모리 슬롯의 메모리 데이터와 상기 N-1(N은 2이상의 자연수) 메모리 덤프 과정에서 기록된 상기 제1 메모리 슬롯의 메모리 덤프 데이터를 XOR 연산한 결과 데이터를 획득할 수 있다.

[0020] 상기 적어도 하나의 프로세서는, 상기 획득된 결과 데이터를 제3 메모리 슬롯에 저장하고, 상기 제3 메모리 슬롯에 저장된 결과 데이터가 상기 제1 메모리 슬롯에 저장되고, 상기 제3 메모리 슬롯에 저장된 결과 데이터에 RLE(Run length Encoder) 압축 알고리즘을 사용하여 압축하고, 상기 압축된 결과 데이터를 스토리지에 저장할 수 있다.

[0021] 상기 적어도 하나의 프로세서는, 상기 수행된 메모리 덤프를 통해 현재 시점의 메모리 덤프 데이터와 현재 시점을 기준으로 이전 시점의 메모리 덤프 데이터의 비교를 통해 변화 데이터를 수집할 수 있다.

[0022] 상기 적어도 하나의 프로세서는, 상기 수집된 변화 데이터에 기초하여 상기 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 상기 IoT 장치의 분석이 요청된 특정 시점을 기준으로 이전 시점의 변화 데이터를 추출하고, 상기 추출된 특정 시점의 변화 데이터와 상기 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 원본 메모리 데이터를 획득할 수 있다.

발명의 효과

[0024] IoT 장치의 기능 및 연산의 일시정지 없이 저지연으로 메모리 덤프를 더욱 빠르게 수행할 수 있으며, 특정 주기마다 연속적으로 메모리를 덤프하여 이전 메모리 덤프 데이터와 현재 메모리 덤프 데이터의 변화를 시계열 단위로 분석하고 변화된 값을 압축 알고리즘을 사용해 저장함으로써 용량을 줄일 수 있다.

[0025] 또한, 특정 시점과 직전 시점의 메모리 변화 값만 불러와 연산을 통해 특정 시점의 메모리 데이터를 바로 획득할 수 있다.

[0026] IoT 장치의 메모리 덤프 과정을 연속적으로 기록함에 따라 IoT 심층 분석 과정에서 이상징후가 발생하던 시점의 메모리로 복원할 수 있으므로 IoT 관리자가 문제 파악을 더욱 빠르게 파악할 수 있다.

[0027] 또한, IoT 관리자가 원격에서 대상 IoT장치의 심층 분석을 수행하고자 할 때 초기 메모리 덤프 파일과 이후 메모리 덤프가 변화된 부분만 전송함으로써, 적은 양의 데이터를 서버에 전송하여 지연을 줄일 수 있다.

[0028] 병렬 처리를 통한 보다 빠른 저지연 메모리 덤프 수행과 메모리 덤프의 델타 값 획득 및 압축을 수행함에 따라 IoT 장치의 문제 파악을 위한 특정 시점의 복원 기능을 통해 IoT 장치를 효율적으로 분석할 수 있다.

도면의 간단한 설명

[0030] 도 1은 일 실시예에 있어서, IoT 장치를 설명하기 위한 도면이다.

도 2는 일 실시예에 있어서, 메모리 덤프 동작을 비교하기 위한 예이다.

도 3 및 도 4는 일 실시예에 있어서, IoT 장치 환경에서의 메모리 덤프 연산 동작을 설명하기 위한 도면이다.

도 5 일 실시예에 있어서, 특정 시점의 메모리 데이터 복구 연산 동작 과정을 설명하기 위한 도면이다.

도 6은 일 실시예에 있어서, 원본 메모리 데이터를 획득하는 동작을 설명하기 위한 도면이다.

도 7은 일 실시예에 있어서, 메모리 연속 수집 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0031] 이하, 실시예를 첨부한 도면을 참조하여 상세히 설명한다.

[0033] 도 1은 일 실시예에 있어서, IoT 장치를 설명하기 위한 도면이다.

[0034] 컴퓨터 시스템은 IoT 장치에서 동작될 수 있으며, IoT 장치의 분석을 위한 저지연 방식의 연속적인 메모리 덤프와 메모리의 변화된 부분 아카이빙 데이터를 통해 특정 시점의 원본 메모리 데이터를 획득할 수 있다.

[0035] 컴퓨터 시스템은 구동 중인 IoT 장치를 대상으로 짧은 시간 안에 메모리 덤프를 수행하기 위하여 병렬 회로에 의해 메모리 덤프를 보다 빠르게 수행할 수 있다. 이러한 과정이 병렬적으로 수행되기 때문에 사용자 및 관리자의 불편을 최소화하여 저연산 및 저지연으로 메모리 덤프가 수행될 수 있다.

[0036] IoT 장치(110)는 프로세서(120), 메모리(130) 및 스토리지(140)로 구성될 수 있다. 메모리(130)는 복수 개의 영역으로 분할될 수 있다. 예를 들면, 메모리(130)는 3개의 영역으로 분할되어, 제1 메모리 슬롯, 제2 메모리 슬롯 및 제3 메모리 슬롯으로 구성될 수 있다.

[0037] 프로세서(120)는 기본적인 산술, 로직 및 입출력 연산을 수행함으로써, 메모리에 포함된 컴퓨터 판독가능한 명령들을 실행하도록 구성된 것일 수 있다. 프로세서(120)는 중앙처리장치, 애플리케이션 프로세서, 또는 커뮤니케이션 프로세서 중 하나 또는 그 이상을 포함할 수 있다. 예를 들면, 프로세서(120)는 IoT 장치(110)의 적어도 하나의 다른 구성요소들의 제어 및/또는 통신에 관한 연산이나 데이터 처리를 실행할 수 있다.

[0038] 메모리(130)는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함하여 구성될 수 있으며, IoT 장치(110)의 적어도 하나의 다른 구성요소에 관계된 명령어 또는 데이터를 저장한다.

[0039] 이때, 제1 메모리 슬롯은 백업 메모리 슬롯(Backup memory slot), 제2 메모리 슬롯은 라이브 IoT 메모리 슬롯(Live IoT memory slot), 제3 메모리 슬롯은 델타 메모리 슬롯(Delta memory slot)을 의미한다.

[0040] 제1 메모리 슬롯은, 백업된 메모리(이전의 메모리 데이터)가 임시로 저장되는 메모리 데이터로 구성되는 임시 저장 공간이고, 제2 메모리 슬롯은 실제 IoT 장치에서 동작하는 펌웨어 등의 시스템이 동작하는 공간이며, 제3 메모리 슬롯은 메모리 데이터 획득을 위한 메모리 덤프를 하는 과정에서 발생하는 XOR 연산으로 인해 발생하는 델타 데이터가 저장되는 공간이다.

[0041] 스토리지(140)는 원형 형태로 구성되어 메모리 덤프와 관련된 메모리 데이터를 저장할 수 있다. 스토리지(140)는 스토리지(140)에 메모리가 모두 저장됨에 따라, 여유 공간이 없을 경우, 오래 저장되어 있던 데이터 순으로 메모리 데이터를 교체하여 새로운 메모리 데이터를 저장할 수 있다.

[0042] 시간에 따라 IoT 장치의 출력값이 변경될 수 있으며, IoT 장치의 메모리를 1tick으로 덤프하기 위한 병렬 메모리 덤프와 이전 시점부터 현재 시점까지 덤프된 메모리 데이터들의 변화될 부분이 계산될 수 있다. 그리고, 이전 시점부터 현재 시점까지 메모리 데이터들의 차이값인 델타 데이터가 제1 메모리 슬롯에 저장될 수 있다. 병

렬 메모리 덤프를 통해 IoT 장치의 메모리값들의 XOR 연산 수행 값이 반환될 수 있으며, 이때, 도출된 차이값인 델타 데이터가 압축되어 스토리지에 저장될 수 있다. 다음 시점으로 넘어간 경우, 스토리지에 저장된 이점 시점 메모리 XOR 데이터인 델타 데이터를 제1 메모리 슬롯을 통해 불러올 수 있다. 이러한 과정을 특정 주기 간격으로 반복함에 따라 메모리의 변화되는 과정을 시계열 기반으로 확인할 수 있다.

[0043] 일 실시예에 따르면, IoT 장치의 분석을 위한 저지연 방식의 메모리 덤프 및 변화를 분석함으로써 종래에 메모리를 분석하는 경우에 발생하는 시스템 멈춤 및 지연이 존재하였던 문제점을 해결할 수 있다.

[0044] 도 2는 일 실시예에 있어서, 메모리 덤프 동작을 비교하기 위한 예이다.

[0045] 도 2(A)는 기존의 메모리 덤프 연산 과정을 나타낸 것이고, 도 2(B)는 실시예에서 제안된 메모리 덤프 연산 과정을 나타낸 것이다. 도 2(A)에 도시된 바와 같이, 메모리 덤프 연산 과정에서 메모리(a)에 있는 메모리 셀을 메모리(b)로 덤프하는데 각 셀이 1 tick에 움직이게 된다. 이에 반해, 도 2(B)에 도시된 바와 같이, 실시예에서 제안된 병렬 처리 메모리 덤프 연산 과정에 의하면 메모리(a)에서 메모리(b)로 셀이 움직이는데 병렬 회로에 의해 1 tick에 전체 덤프가 가능해진다.

[0046] 이와 같은 과정으로 IoT 장치가 연산 수행 중에도 덤프를 위해 연산 수행을 중단하거나 지연되는 상황이 상대적으로 저지연으로 수행될 수 있으며 이를 통해 IoT 장치 메모리 덤프 수집을 보다 빠르게 연속적으로 수집할 수 있게 된다.

[0047] 도 3은 일 실시예에 있어서, IoT 장치 환경에서의 메모리 덤프 연산 동작을 설명하기 위한 도면이다.

[0048] 컴퓨터 시스템은 IoT 장치의 메모리 데이터 수집 및 모니터링을 위하여 FPGA 기반의 병렬 회로를 이용한 초고속 메모리 덤프 기술을 사용할 수 있다. 도 3은 CPU, 메모리, 스토리지로 구성된 IoT 장치 환경에서, IoT 장치의 메모리에서 메모리 덤프를 수행하는 과정을 나타낸 것이다.

[0049] 도 3을 참고하면, IoT 장치의 메모리 영역이 제1 메모리 슬롯, 제2 메모리 슬롯 및 제3 메모리 슬롯으로 구성된 것을 나타낸 것이다.

[0050] 도 4를 참고하면, IoT 장치의 메모리 영역이 제2 메모리 슬롯 및 복수 개의 제3 메모리 슬롯으로 구성된 것을 나타낸 것이다. 도 4에서는 제2 메모리 슬롯 및 2개의 제3 메모리 슬롯으로 구성된 것을 도시하였다. 이때, 복수 개의 제3 메모리 슬롯 중 어느 하나의 제3 메모리 슬롯은 도 3의 제1 메모리 슬롯의 역할을 수행하게 된다. 예를 들면, 복수 개의 제3 메모리 슬롯 중 어느 하나의 제3 메모리 슬롯이 선택될 수 있다. 예를 들면, 시간이 n(n은 자연수 또는 양의정수)라고 가정하기로 한다. n이 짝수일 때 나머지가 0이므로 제3 메모리 슬롯(M3)을 사용하고, n이 홀수일 때 나머지가 1이므로 제1 메모리 슬롯(M1)을 사용할 수 있다. 메모리 슬롯을 번갈아 사용하도록 하여 이전 메모리 값과 XOR이 되도록 할 수 있다. 이때, 메모리 덤프가 처음 수행되었을 경우와 메모리 덤프가 이미 수행된 적 있는 경우에 따라 제3 메모리 슬롯이 다르게 선택될 수 있다. 도 3과 도 4에서 메모리 덤프를 수행하는 과정은 동일하다.

[0051] 첫 번째 메모리 덤프가 수행됨에 따라 제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 모든 데이터가 0으로 초기화될 수 있다. 그 후, 제2 메모리 슬롯에 저장되어 있던 메모리 데이터들이 FPGA 기반의 병렬 회로를 통해 제1 메모리 슬롯으로 빠르고 전달될 수 있다. 다음 메모리 덤프가 수행됨에 따라 현재 동작 중인 메모리인 제2 메모리 슬롯의 메모리 데이터와 이전 메모리 덤프 동작 과정에서 기록된 제1 메모리 슬롯의 메모리 덤프 데이터에 대하여 XOR 연산(①)을 수행한 델타 데이터(②)가 제3 메모리 슬롯에 저장될 수 있다. 제3 메모리 슬롯의 델타 데이터들은 제1 메모리 슬롯에 저장되며 다음 메모리 데이터 값과 XOR 연산에 사용될 수 있다. 그 후, 제3 메모리 슬롯에 저장되었던 델타 데이터들이 IoT 장치를 통해 RLE(Run-Length Encoding)(③) 과정을 거쳐 스토리지에 저장될 수 있다. 이때, 델타 데이터는 실시간 IoT 장치의 메모리 덤프 데이터와 이전 XOR 결과 데이터를 XOR하여 이전 결과에 더해지는 방식으로 스토리지에 저장될 수 있다. 스토리지에 저장된 델타 데이터가 IoT 모니터링 서버로 전달될 수 있다. 이와 같은 방식은 분석을 필요로 하는 시점의 메모리 복원 과정에서 연산을 단축시킬 수 있다. 특정 시점의 메모리 복원을 위해서 전체 XOR 데이터를 사용하지 않고 복원하려는 시점의 델타 데이터와 특정 시점을 기준으로 바로 직전 시점의 델타 데이터만으로 XOR 연산을 수행하여 보다 간략한 연산으로 메모리 복원이 수행될 수 있다. 서버에서 원하는 시점의 메모리를 RLE를 통하여 압축을 풀어 복원을 위한 XOR이 수행될 수 있다.

[0052] RLE 과정은 리소스가 충분하지 않은 IoT 장치 환경에서 데이터를 최소화하기 위한 과정으로써, 제2 메모리 슬롯과 제1 메모리 슬롯의 XOR 연산을 통해, 시스템에 의해 변경된 메모리 부분이 추출됨으로써 제3 메모리 슬롯은 그 특성상 중복된 데이터의 양이 많을수록 RLE(③) 과정을 통해 데이터의 크기를 최소화할 수 있다. 상기 설명

한 바와 같이, 구동 중인 IoT 장치의 제2 메모리 슬롯과 이전 XOR 결과인 델타 데이터가 저장된 제1 메모리 슬롯이 병렬 회로에 의해 메모리 덤프가 수행되고, 병렬 회로에 의해 XOR 연산이 수행될 수 있다. 이러한 과정이 반복되어 IoT 장치의 메모리가 연속적으로 기록될 수 있다.

[0053] 도 5 일 실시예에 있어서, 특정 시점의 메모리 데이터 복구 연산 동작 과정을 설명하기 위한 도면이다.

[0054] 사용자가 모니터링 환경을 통해 IoT 장치의 특정 시점의 메모리 데이터를 복구하고자 하는 경우, IoT 장치의 스토리지에 저장된 메모리 덤프 데이터가 RLE 과정을 통해 결과 데이터(델타 메모리(delta-memory) 값)로 복원될 수 있다. 복원된 결과 데이터(델타 메모리 값)는 현재 시스템에서 사용되고 있는 제2 메모리 슬롯의 메모리 데이터와 XOR 연산을 수행하게 되고, XOR 연산의 결과로서 가장 최근 덤프된 메모리의 이전 시점의 메모리 덤프 데이터로 복원될 수 있다. 복원 과정에서 생성된 델타 데이터를 대상으로 XOR 연산을 특정 횟수를 반복하게 되면, 사용자가 원하는 시점의 메모리 덤프 데이터가 복원될 수 있다.

[0055] 동작 중인 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프 연산을 통해 전체 메모리 데이터를 한 클럭에 복제를 수행하고 이전 주기의 메모리와 XOR 연산을 수행해 나온 델타 값을 시간 축을 기준으로 순차적으로 저장함으로써 시간에 따라 메모리가 변화하는 모습을 확인할 수 있다.

[0056] 일례로, 메모리 덤프 수행 과정을 통해 분석하고자 하는 대상 IoT 장치가 각 IoT 장치의 기능을 수행하는 과정에서 메모리 분석이 필요한 시점에 메모리 덤프 과정을 통해 IoT 장치의 기능 및 연산의 일시정지 없이 저지연으로 메모리 덤프를 더욱 빠르게 수행할 수 있다. 특정 주기마다 메모리를 덤프하여 이전 메모리 덤프 데이터와 현재 메모리 덤프 데이터의 변화 데이터를 시계열 단위로 분석하고, 변화 데이터를 RLE 기반의 압축 알고리즘을 사용하여 저장함으로써 용량을 줄일 수 있다.

[0057] 사용자가 메모리 복구 과정을 통해 특정 시점의 메모리 데이터를 얻기 위해서는 특정 시점과 직전 시점의 메모리 변화 데이터(값)만 불러와 연산을 통해 바로 획득할 수 있다. 이러한 과정을 통해 IoT장치의 메모리 덤프 과정을 연속적으로 신속하게 기록함에 따라 IoT 심층 분석 과정에서 이상징후가 발생하던 시점의 메모리로 복원할 수 있으므로 IoT장치의 문제를 더욱 빠르게 파악할 수 있다.

[0058] IoT 플랫폼 사용자가 원격에서 대상 IoT의 심층 분석을 수행하고자 할 때, 초기 메모리 덤프 파일과 이후 메모리 덤프가 변화된 적은 양의 메모리 덤프 데이터를 서버에 전송하여 지연을 최소화할 수 있다. 병렬 회로를 통한 초고속 메모리 덤프 수행과 메모리 덤프의 델타 값 획득 및 압축 과정을 통해 IoT 장치의 이상행위 파악을 위한 특정 시점 복원 기능으로 IoT 장치 및 IoT 시스템 분석에 있어 효율성과 경제성이 향상될 수 있다.

[0059] 도 5를 참고하면, IoT 장치의 메모리를 0으로 초기화하여 시작할 수 있다. IoT 장치가 구동되면 연산이 수행되어 쓰이는 메모리는 병렬 회로에 의해 덤프를 수행할 수 있다. 시작 과정에서 제1 메모리 슬롯은 0으로 초기화되어있다. 구동 중인 IoT 장치의 제2 메모리 슬롯과 제1 메모리 슬롯은 병렬 회로에 의해 XOR 연산이 수행되고, 연산을 수행한 결과 데이터(델타 값)가 제3 메모리 슬롯에 저장될 수 있다. 이때, 구동중인 IoT 장치를 대상으로 짧은 시간 안에 메모리 덤프를 수행하기 위해 병렬 접근 회로를 사용하여 메모리 덤프가 빠르게 수행될 수 있다. 첫 메모리 덤프가 아닌 경우 스토리지에 존재하는 메모리 변화 데이터를 RLE를 통해 불러오고, 제1 메모리 슬롯에 존재하는 메모리 덤프 데이터와 현재 시점에 덤프된 메모리 데이터를 XOR 연산을 수행한 후 제3 메모리 슬롯에 저장 후, RLE를 통해 압축할 수 있다. 압축된 결과 데이터가 스토리지에 순차적으로 저장될 수 있다. 제3 메모리 슬롯의 결과 데이터(델타 값)은 다시 제1 메모리 슬롯에 저장되며 다음 IoT 장치의 메모리 값과 XOR에 사용될 수 있다.

[0060] 그리고 다시 구동중인 제2 메모리 슬롯의 메모리 데이터와 이전 XOR 결과인 델타가 저장된 제1 메모리 슬롯이 병렬 회로에 의해 덤프가 수행되고 병렬 회로에 의해 XOR 연산이 수행될 수 있다. 해당 과정이 반복되어 IoT 메모리가 연속적으로 기록될 수 있다.

[0061] 결과 데이터는 실시간 IoT 메모리 덤프 데이터와 이전 XOR 결과 값에 XOR하여 이전 결과 데이터에 더해지는 방식으로 스토리지에 저장될 수 있다. 이와 같은 방식은 분석을 필요로 하는 시점의 메모리 복원 과정에서 연산을 단축시킬 수 있다. 특정 시점의 메모리 복원을 위해서 전체 XOR 데이터를 사용하지 않고 복원하려는 시점의 델타 데이터와 바로 직전 시점의 델타 데이터만으로 XOR 연산을 수행하여 보다 간략한 연산으로 메모리 복원을 수행할 수 있다.

[0062] 도 6은 일 실시예에 있어서, 원본 메모리 데이터를 획득하는 동작을 설명하기 위한 도면이다.

[0063] 변화 데이터에 기초하여 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 IoT 장치의 분석이 요청된 특정

시점을 기준으로 이전 시점의 변화 데이터가 추출될 수 있다. 추출된 특정 시점의 변화 데이터와 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 메모리 데이터가 획득될 수 있다.

[0064] 일례로, 스토리지에 특정 시점(t_n)과 특정 시점을 기준으로 바로 직전 시점(t_{n-1}) 시점의 메모리 데이터가 추출될 수 있다.

$$\delta_{tn-1} = 0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \cdots \oplus S_{tn-1}$$

$$\delta_{tn} = 0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \cdots \oplus S_{tn-1} \oplus S_{tn}$$

[0065]

[0066] 추출된 메모리 데이터에서 중복되는 영역이 소거될 수 있다.

$$\delta_{tn-1} = \boxed{0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \quad \oplus S_{tn-1}}$$

$$\delta_{tn} = \boxed{0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \cdots \oplus S_{tn-1}} \oplus S_{tn}$$

[0067]

[0068] 특정 시점(t_n)의 메모리 데이터가 획득될 수 있다.

$$\delta_{tn} = 0 \oplus S_{tn} = S_{tn}$$

[0069]

[0070] 실시예에 따르면, 현시점의 메모리 데이터를 기준으로 역으로 XOR 연산을 수행해 이전 주기의 원본 메모리 데이터를 구하는 과정을 반복하여 특정 시점의 메모리 데이터를 구할 수 있어 동작 중인 IoT 장치에 대해 심층적인 분석을 용이하게 할 수 있다.

[0071] 도 7은 일 실시예에 있어서, 메모리 연속 수집 방법을 설명하기 위한 흐름도이다.

[0072] 단계(710)에서 컴퓨터 시스템은 구동 중인 IoT 장치를 대상으로 IoT 장치에 구성된 병렬 회로 기반의 메모리 구조를 이용하여 메모리 덤프를 수행함에 따라 각 메모리 덤프 주기 간 발생하는 메모리 덤프 데이터의 변화 데이터를 수집할 수 있다. 컴퓨터 시스템은 IoT 장치에 구성된 메모리에 메모리 덤프 연산을 위한 복수 개의 슬롯을 생성하고, 생성된 복수 개의 슬롯을 이용하여 IoT 장치의 메모리 데이터를 덤프 주기에 따라 병렬화된 메모리 덤프를 수행할 수 있다. 컴퓨터 시스템은 제1 메모리 슬롯과 제3 메모리 슬롯에 위치한 메모리 데이터를 0으로 초기화하고, 제1 메모리 덤프를 수행할 경우, FPGA 기반의 병렬 회로를 통해 제2 메모리 슬롯에 저장된 메모리 데이터를 0으로 초기화된 제1 메모리 슬롯으로 전달할 수 있다. 컴퓨터 시스템은 제1 메모리 덤프를 수행한 이후, 제N (N은 2이상의 자연수) 메모리 덤프를 수행할 때, 제N (N은 2이상의 자연수) 메모리 덤프 과정에서의 제2 메모리 슬롯의 메모리 데이터와 N-1(N은 2이상의 자연수) 메모리 덤프 과정에서 기록된 제1 메모리 슬롯의 메모리 덤프 데이터를 XOR 연산한 결과 데이터를 획득할 수 있다. 컴퓨터 시스템은 획득된 결과 데이터를 제3 메모리 슬롯에 저장하고, 제3 메모리 슬롯에 저장된 결과 데이터가 제1 메모리 슬롯에 저장되고, 제3 메모리 슬롯에 저장된 결과 데이터에 RLE(Run length Encoder) 압축 알고리즘을 사용하여 압축하고, 압축된 결과 데이터를 스토리지에 저장할 수 있다. 컴퓨터 시스템은 수행된 메모리 덤프를 통해 현재 시점의 메모리 덤프 데이터와 현재 시점을 기준으로 이전 시점의 메모리 덤프 데이터의 비교를 통해 변화 데이터를 수집할 수 있다.

[0073] 단계(720)에서 컴퓨터 시스템은 수집된 변화 데이터에 기초하여 IoT 장치의 분석이 요청된 특정 시점의 원본 메모리 데이터를 획득할 수 있다. 컴퓨터 시스템은 수집된 변화 데이터에 기초하여 IoT 장치의 분석이 요청된 특정 시점의 변화 데이터와 IoT 장치의 분석이 요청된 특정 시점을 기준으로 이전 시점의 변화 데이터를 추출하고, 추출된 특정 시점의 변화 데이터와 특정 시점을 기준으로 이전 시점의 변화 데이터에서 중복되는 영역을 소거함으로써 특정 시점의 메모리 데이터를 획득할 수 있다.

[0074] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령

(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0075] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

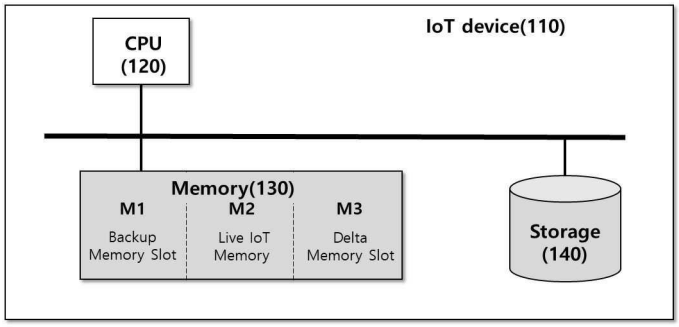
[0076] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

[0077] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

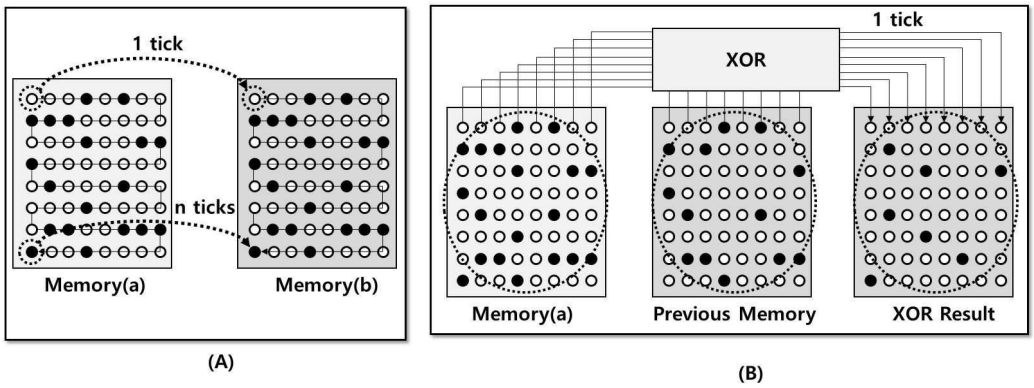
[0078] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

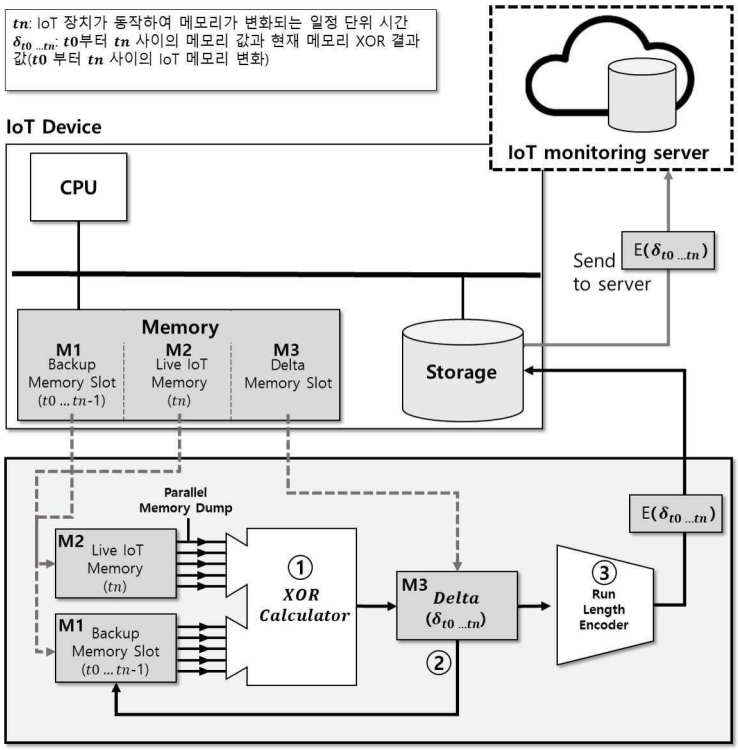
도면1



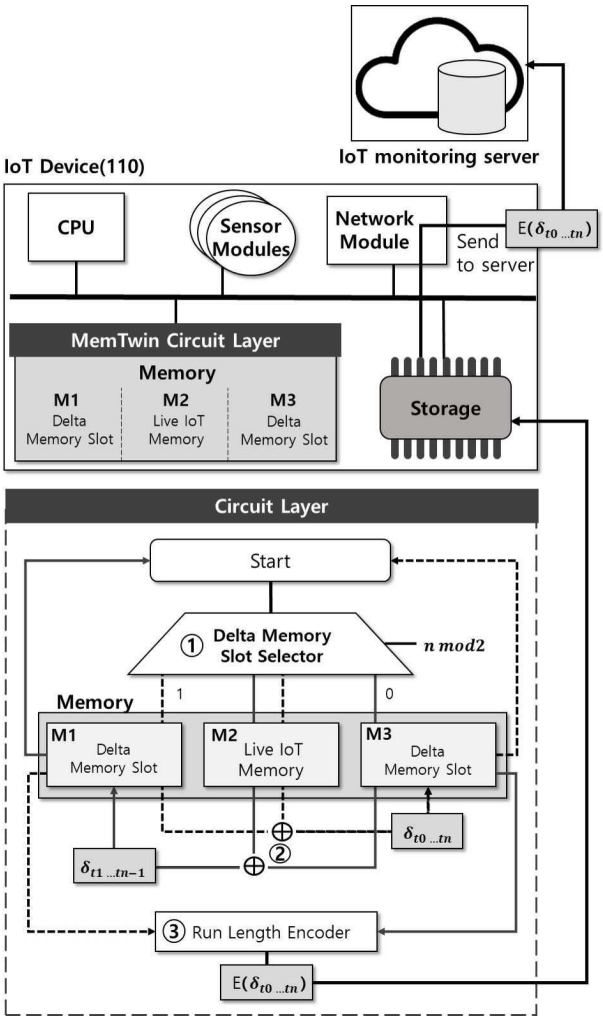
도면2


1 tick
n ticks
Memory(a)
Memory(b)
(A)
1 tick
XOR
Memory(a)
Previous Memory
XOR Result
(B)

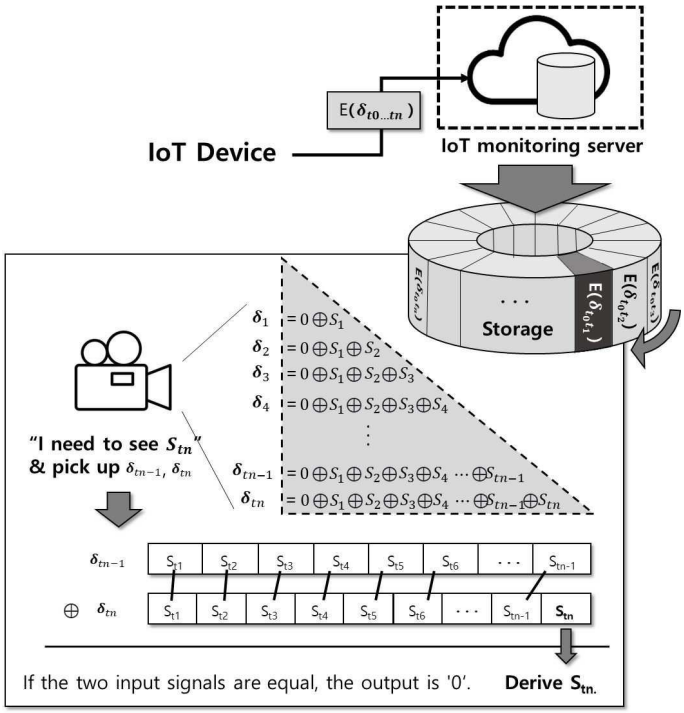
도면3


tn: IoT 장치가 동작하여 메모리가 변화되는 일정 단위 시간
δt0 ...tn: t0부터 tn 사이의 메모리 값과 현재 메모리 XOR 결과 값(t0 부터 tn 사이의 IoT 메모리 변화)
IoT Device
CPU
IoT monitoring server
Send to server
E(δt0 ...tn)
Memory
M1 Backup Memory Slot (t0 ... tn-1)
M2 Live IoT Memory (tn)
M3 Delta Memory Slot
Storage
Parallel Memory Dump
M2 Live IoT Memory (tn)
M1 Backup Memory Slot (t0 ... tn-1)
① XOR Calculator
M3 Delta (δt0 ...tn)
②
③ Run Length Encoder
E(δt0 ...tn)

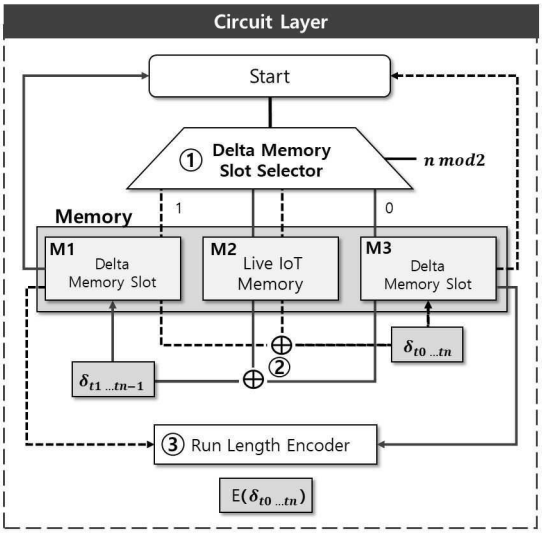
도면4


IoT monitoring server
IoT Device(110)
CPU
Sensor Modules
Network Module
Send to server
$E(\delta_{t0\ldots tn})$
MemTwin Circuit Layer
Memory
M1 Delta Memory Slot
M2 Live IoT Memory
M3 Delta Memory Slot
Storage
Circuit Layer
Start
① Delta Memory Slot Selector
$n\ mod 2$
1
0
Memory
M1 Delta Memory Slot
M2 Live IoT Memory
M3 Delta Memory Slot
$\delta_{t0\ldots tn}$
$\delta_{t1\ldots tn-1}$
②
③ Run Length Encoder
$E(\delta_{t0\ldots tn})$

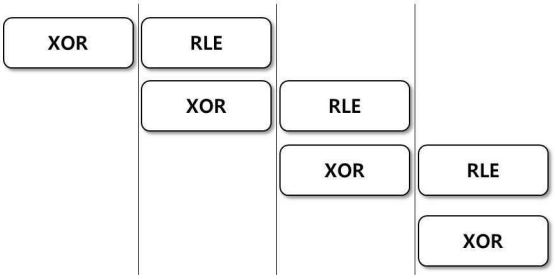
도면5


$E(\delta_{t0\ldots tn})$
IoT Device
IoT monitoring server
Storage
$E(\delta_{t0t1})$
$\delta_1 = 0 \oplus S_1$
$\delta_2 = 0 \oplus S_1 \oplus S_2$
$\delta_3 = 0 \oplus S_1 \oplus S_2 \oplus S_3$
$\delta_4 = 0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4$
$\delta_{tn-1} = 0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \cdots \oplus S_{tn-1}$
$\delta_{tn} = 0 \oplus S_1 \oplus S_2 \oplus S_3 \oplus S_4 \cdots \oplus S_{tn-1} \oplus S_{tn}$
"I need to see S_{tn}"
& pick up δ_{tn-1}, δ_{tn}
δ_{tn-1}
$\oplus$ δ_{tn}
S_{t1} S_{t2} S_{t3} S_{t4} S_{t5} S_{t6} ... S_{tn-1}
S_{t1} S_{t2} S_{t3} S_{t4} S_{t5} S_{t6} ... S_{tn-1} S_{tn}
If the two input signals are equal, the output is '0'.
Derive S_{tn}.

도면6



tn	1	2	3	~	n
M2 (S_{tn})	S_1	S_2	S_3	~	S_{tn}
M1	$0 \oplus S_1$	$0 \oplus S_1$	$0 \oplus S_1 \oplus S_2 \oplus S_3$	~	$0 \oplus S_1 \oplus S_2 \sim S_{tn}$
M3	0	$0 \oplus S_1 \oplus S_2$	$0 \oplus S_1 \oplus S_2$	~	$0 \oplus S_1 \oplus S_2 \sim S_{tn-1}$



도면7

