ISSN : 1226-7244 (Print)
ISSN : 2288-243X (Online)
논문번호 23-02-06
j.inst.Korean.electr.electron.eng.Vol.27,No.2,165~171,June 2023
https://doi.org/10.7471/ikeee.2023.27.2.165
39

소프트 에러로부터 자동 복구가 가능한 삼중 코어 지연 락스텝 프로세서의 설계

Design of a Delayed Triple-Core Lock-Step Processor with Auto-Recovery from Soft Errors

양 성 현*, 김 주 호*, 이 성 수*★

Seonghyun Yang*, Juho Kim*, and Seongsoo Lee*★

Abstract

Soft error due to radiation induces temporary errors in semiconductors, and it causes fatal danger in the automotive applications. Delayed triple-core lock-step processor performs identical operations in three independent cores with different delays. Its final result is determined by voting results of three cores, so it is strongly resilient to soft error. However, the processor cannot recover its error-occured core and stops it, and the processor continues its operation with remaining two cores. When additional soft error occurs again, the processor can recognize that an error has occurred in one of remaining two cores, but it cannot recognize which core has an error, so the entire processor must be initialized. This paper proposes an auto-recovery delayed triple-core lock-step processor that continues its operation without initialization even if multiple soft errors occur. It recovers its error-occured core by overwriting internal values of the normal core into the error-occured core. Simulation results show the processor was successfully recovered without initialization and performed its operation continuosly when soft errors occurred several times.

요 약

방사선 등으로 반도체 내부에서 일시적으로 에러가 발생하는 소프트 에러는 차량용 어플리케이션에서 심각한 위험을 초래한다. 동일한 작업을 세 개의 코어에서 서로 다른 지연 시간을 가지고 수행하여 다수결로 결과를 비교하여 최종 결과를 출력하는 삼중 코어 지연 락스텝 프로세서는 소프트 에러에 매우 강인하지만 에러가 발생한 코어의 내부 값을 복구하는 기능이 없어서 해당 코어를 정지시키고 나머지 두 코어만으로 동작을 수행한다. 이 때 추가로 소프트 에러가 발생하면 두 코어의 결과값을 비교하여 에러 발생은 알 수 있지만 어느 코어인지 알 수 없어서 프로세서 전체를 초기화하고 다시 동작을 수행해야 한다. 본 논문에서는 정상 코어의 내부 값을 에러가 발생한 코어에 덮어 씌워서 코어를 복구하는 자동 복구 삼중 코어 지연 락스텝 프로세서를 제안한다. 시뮬레이션 결과 소프트 에러가 여러 번 발생해도 초기화 없이 자동 복구되어 연속 동작을 수행하였다.

Key words : Error Tolerant, Delayed Lock-Step, Triple Modular Redundancy, Triple-Core Processor, System Reset, Core Recovery, Soft Error, Processor Design

* Soongsil University (Master Student, Professor)
★ Corresponding author
E-mail : sslee@ssu.ac.kr, Tel : +82-2-820-0692

※ Acknowledgment
This work was supported by the R&D Program of the Ministry of Trade, Industry, and Energy (MOTIE) (20012624, RS-2022-00155731, RS-2023-00232192).
Manuscript received Jun. 21, 2023; revised Jun. 23, 2023; accepted Jun. 25, 2023.

Ⅰ. 서론

자동차의 디지털화가 가속화됨에 따라, 자동차를 제어하는 ECU(Electronic Control Unit)의 역할이 점차 중요해지고 있다. 이 ECU는 차량의 동작을 제어하는 핵심 역할을 수행하므로, 차량의 안전성에 큰 영향을 미친다. 이러한 이유로 자동차 기능 안전에 대한 국제 표준인 ISO 26262[1]에서는 ECU를 ASIL(Automotive Safety Integrity Level)에 의거하여 엄격하게 관리하고 있다[2]. 이 ASIL 규정을 만족하기 위해서 대다수의 ECU는 에러 처리, 하드웨어 중복, 신뢰성 향상 등 다양한 기술을 적용하고 있다.

ECU는 기능 안전성을 보장하기 위해, 에러 처리 기술 중 하나인 락스텝(Lock-Step)[3]을 활용하고 있다. 락스텝은 다수의 프로세서 코어에 동일한 입력값을 제공하고 동일한 작업을 수행하게 함으로써, 한 코어에서 에러가 발생해도 시스템 전체는 정상 작동을 유지할 수 있도록 하는 기술이다.

그러나, 소프트 에러는 전자기파, 방사선 등의 외부 요인으로 인해 발생하며, 이러한 에러의 발생원에 따라 짧은 시간 동안 광범위한 영역에 영향을 미칠 수 있다. 이런 특성으로 인해, 일반적인 락스텝 방식이 적용된 프로세서에서도 여러 코어가 동시에 에러를 발생시키는 공통 모드 고장(Common Mode Failure)의 위험이 존재한다. 이런 문제를 해결하기 위해 지연 락스텝(Delayed Lock-Step)과 같은 기술이 주목받고 있다. 이 기술은 두 개 이상의 코어에서 시간차를 두고 동일한 연산을 수행하는 방식을 사용하여 공통 모드 고장을 방지하는 데 큰 도움을 주는데, 특히 세 개 이상의 코어를 사용하는 삼중 코어 지연 락스텝(D-TCLS : Delayed Triple-Core Lock-Step) 방식이 연구되었다[4][5].

하지만 D-TCLS에서도 코어에서 에러가 발생하면 해당 코어를 복구하지 못해 동작을 정지시키고 나머지 두 코어로 시스템을 계속 작동시키는 한계를 가지고 있다. 만약 추가적인 에러가 발생하면 어떤 코어에서 에러가 발생했는지 판별할 수 없기 때문에 시스템 전체를 초기화하는 방법을 택한다. 이러한 초기화 과정은 시스템이 다시 정상적으로 부팅되어 동작할 때까지의 지연 시간을 필요로 하므로 안전에 위협이 될 수 있다.

이러한 문제를 해결하기 위해, 본 논문에서는 정상 코어의 내부 값을 에러가 발생한 코어에 덮어 씌움으로서 정상화시키는 자동 복구 삼중 코어 지연 락스텝 프로세서를 제안한다. 제안하는 프로세서는 에러가 발생한 코어를 성공적으로 복구하고 다시 투표 대열에 편입시켜 시스템 초기화가 필요 없이 D-TCLS 모드로 작동하게 하여 지속적이고 안정적인 시스템 운용이 가능하도록 한다. 제안하는 프로세서를 Verilog HDL로 설계하였으며 IDEC(IC Design Education Center)에서 제공한 Siemens Modelsim을 이용하여 시뮬레이션으로 검증하였다.

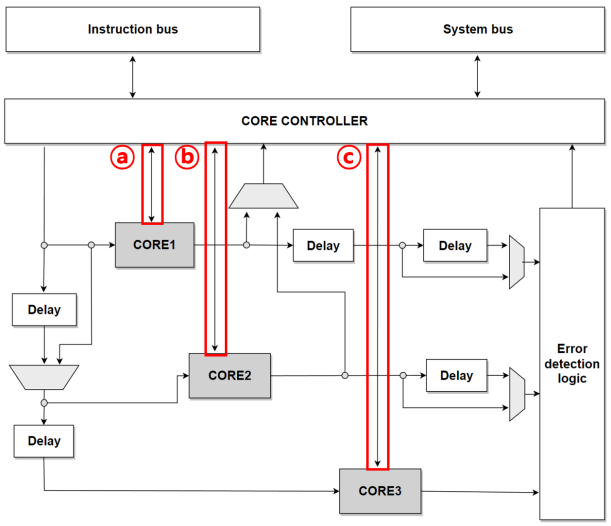

Fig. 1. Architecture of the automatic recovery delayed triple-core lock-step processor.
그림 1. 자동 복구 삼중 코어 지연 락스텝 프로세서의 구조

Ⅱ. 제안하는 프로세서의 아키텍쳐

그림 1은 제안하는 자동 복구 D-TCLS 프로세서의 구조를 보여준다. ARM Cortex-M3 프로세서 코어[6]와 AMBA AHB-LITE 버스를 이용하여 시스템을 설계하였으며 각 코어의 입력과 출력에 지연 요소를 적용하고, 이를 제어하기 위한 복수 선택기(MUX)를 도입하였다. 코어의 복구를 위해 각 코어의 일반 목적 레지스터(GPR) 및 특수 목적 레지스터(SPR) 값을 코어 제어기로 전송하는 데이터 라인, 복구 타이밍을 위한 Flush Detect 라인 등 코어의 복구에 필요한 데이터 및 제어 신호의 연결을 ⓐ, ⓑ, ⓒ와 같이 코어 제어기와 각 코어 사이에 구현하였다.

1. 복구 타이밍을 위한 Flush Detect 라인 추가

제안하는 프로세서에서 개별 코어로 사용한 ARM Cortex-M3는 3단계 파이프라인으로 동작한다. 코어 내부의 구성으로 IFU(Instruction Fetch Unit), IDU (Instruction Decode unit), EXU(Execute operation

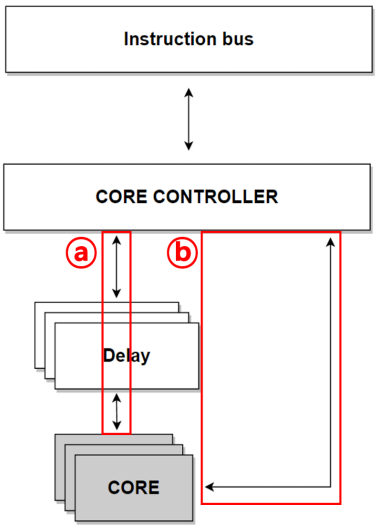

Fig. 2. Sub-core instruction input and recovery paths.
그림 2. 서브 코어 인스트럭션 입력 및 복구 경로

Unit), LSU(Load/Store Unit)이 있다. 이때, 에러가 발생한 코어를 정상화하기 위해서는 모든 파이프라인이 정상 코어와 값을 공유해야 하는데, 이렇게 시스템을 구성하게 되면 코어 사이에 연결선이 매우 많아져 시스템의 복잡도를 크게 증가시키게 된다.

본 연구는 코어 복구에 사용되는 연결선의 사용을 최소화하기 위해 Branch 명령어를 통해 내부 파이프라인을 초기화하는 Flush 타이밍에서 일반 목적 레지스터(GPR) 값 및 특수 목적 레지스터(SPR) 값만을 사용하여 코어를 정상화하는 방법을 도입하였다. 또한, 정상 코어가 Branch 명령어를 통해 Flush 되는 시점을 감지하기 위해 코어 내부의 IDU에 Flush Detect 라인을 추가하였고, 이 라인을 코어 제어기에 연결하였다.

2. 인스트럭션 버스와 코어의 직접 연결을 위한 멀티플렉서 추가

코어가 초기화될 때 일반 목적 레지스터(GPR)의 링크 레지스터, 스택 포인터, 프로그램 카운터 값을 초기화하고, 리셋 벡터를 통해 시스템이 정상적으로 동작하기 시작한다[6]. 그러나 에러가 발생한 코어를 복구하려면 초기화 과정 중 인스트럭션 데이터를 코어로 적절하게 전달해야 한다는 점이 중요하다. 이 데이터는 코어가 정상적인 동작을 시작할 수 있도록 리셋 벡터와 연결되어 있다.

본 시스템은 메인 코어(그림 1의 Core1)의 입력을 단순히 지연하여 전달 받는 구조로 두 개의 서브 코어(그림 1의 Core2, Core3)가 설계되어 있다. 각 코어가 독립적으로 초기화 과정을 수행하려면 벡터 테이블과 같은 필수 데이터에 각 코어가 독립적으로 접근할 수 있어야 한다. 이를 위해 본 연구에서는 코어 제어기 내부의 MUX를 통해 메인 코어가 아닌 코어도 메인 시스템 버스에 직접 접근할 수 있게 설계하였다.

보통 상태에서 서브 코어의 인스트럭션 입력은 그림 2의 ⓐ를 통해 데이터가 이동한다. 그러나 서브 코어가 복구 상태에 들어가면 인스트럭션 버스에 직접 접근이 가능해야 하므로, 코어 제어기의 MUX에 의해 ⓑ를 통해 시스템 인스트럭션 버스에 접속하게 설계하였다. 따라서, 벡터 데이터는 초기화 중인 코어가 인스트럭션 버스를 통해 직접 접근할 수 있게 설계되었다. 이 설계를 통해 코어의 독립적인 초기화가 가능해진다.

3. 코어 복구를 위한 GPR 및 SPR 수정

코어가 정상적으로 작동하기 위해서는 GPR과 SPR의 값을 외부 신호를 통해 수정할 수 있어야 한다. 이를 가능하게 하도록, 본 연구에서는 코어 제어기로부터의 신호가 각 코어 내부의 GPR 및 SPR의 값을 읽고 변경할 수 있도록 GPR과 SPR을 수정하였다.

4. 복구 명령어 패치를 위한 인스트럭션 입력기 설계

프로그램 카운터(PC)를 링크 레지스터 (LR=R14)로 복사해주기 위해 Thumb 명령어 세트의 'BX LR' 명령어를 사용한다. 'BX LR'는 Branch 명령어의 한 종류로, LR에 저장된 리턴 주소를 PC로 이동시키는 역할을 한다. 이 명령어가 복구가 필요한 코어에 패치되게 하기 위해 코어 제어기 내부에 'BX LR'명령어가 복구가 필요한 코어에 패치될 수 있게 해당 코어의 인스트럭션 데이터 입력으로 출력하도록 설계한다.

Ⅲ. 제안하는 프로세서의 동작

제안하는 자동 복구 D-TCLS 프로세서는 그림 3의 순서도에 따라 동작한다. 기존 D-TCLS 프로세서는 D-TCLS 모드로 시스템이 동작하다가 에러가 발생하면 해당 코어를 정지시키고 두 개의 코어만으로 동작하는 D-DCLS (Delayed Dual-Core Lock-Step) 모드로 시스템이 동작한다. 이때 메인 코어에서 에러가 발생하면 메모리 Roll-Back과 메모리 Write-Back을 통해 에러가 발생한 코어의 작동을 보정한 뒤 서브 코어를 메인 코어로 전환해주며, 서브 코어에서 에러가 발생하였으면 단순히 에러가 발생한 서브 코어를 작동 중지시킨다.

기존 연구[4][5]에서는 D-TCLS 모드에서 에러가 발생하여 D-DCLS 모드로 동작하다가 또 다시 에러가 발생

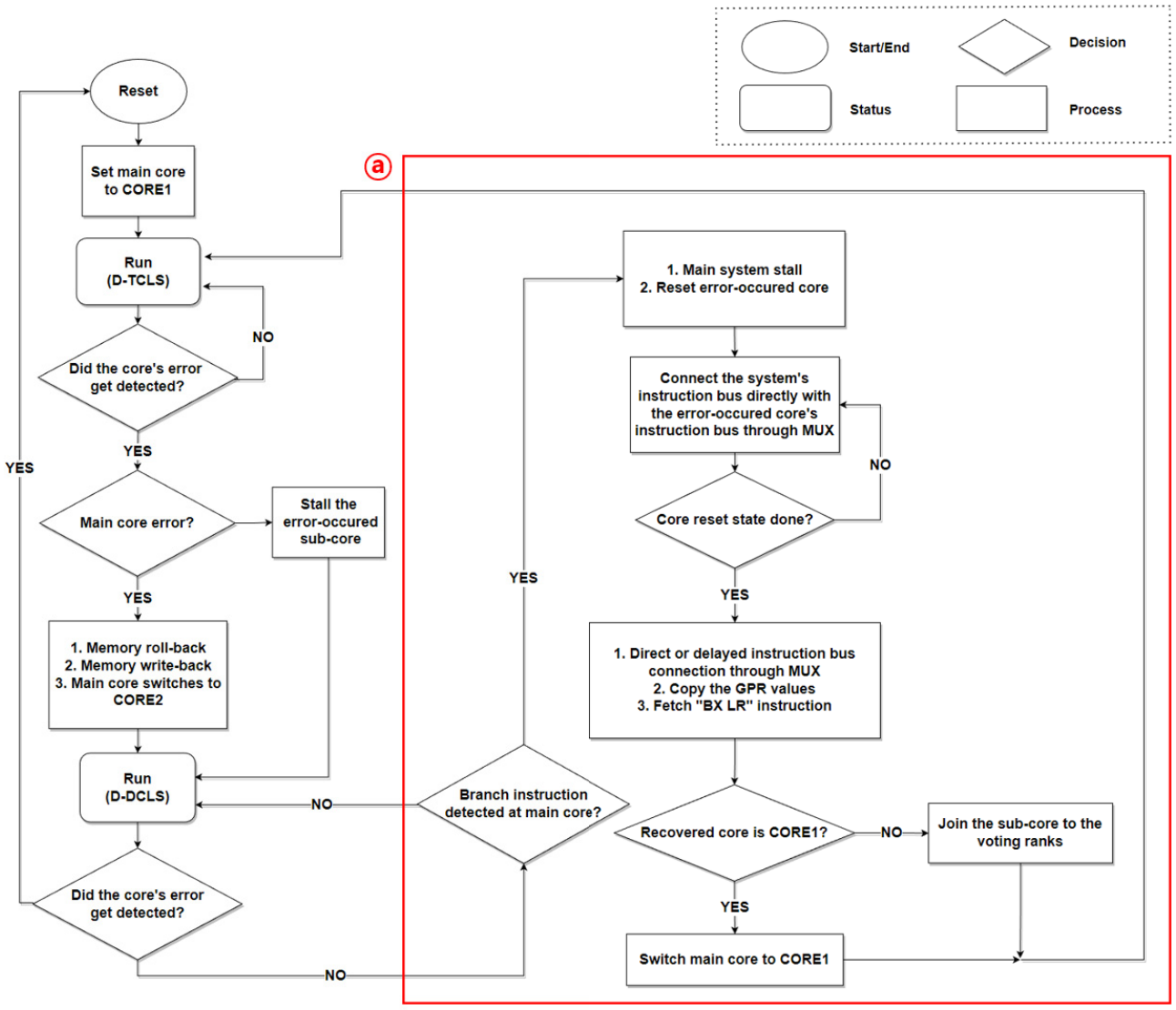

Fig. 3. Operational flowchart of the automatic recovery delayed triple-core lock-step processor.
그림 3. 자동 복구 삼중 코어 지연 락스텝 프로세서의 동작 순서도

하면 전체 시스템을 초기화시켜 시스템을 정상화하였다. 본 논문에서는 에러가 발생했을 때 그림 3과 같이 일단 D-DCLS 모드에 진입하지만 메인 코어에서 Branch 명령어가 수행되는 시점에서 에러가 발생한 코어를 복구하여 다시 D-TCLS 모드로 전환하므로 초기화 없이 연속적으로 동작이 가능하다. 다만 메인 코어에서 Branch 명령어가 수행되기 전에 D-DCLS 모드로 동작하는 동안 또다시 에러가 발생하면 부득이하게 시스템을 초기화해야 하지만 D-DCLS 모드에 머무르는 시간은 전체 동작 시간 중에서 극히 짧기 때문에 현실적으로는 거의 초기화하지 않아도 된다고 볼 수 있다.

코어 에러 복구 과정은 그림 3의 ⓐ에 표현되어 있다. 이 과정은 코어 에러를 효과적으로 복구하며, 각 코어가 에러가 발생한 후에도 복구과정을 통한 뒤, D-TCLS 타이밍을 유지하도록 해준다. 각 단계의 세부 사항은 아래에서 상세히 설명한다.

1. 코어 초기화 과정

먼저 메인 코어에서 분기 명령어가 탐지되면 전체 파이프라이닝을 맞추기 위해 메인 시스템을 잠시 정지시키며 정상화가 필요한 에러 코어를 초기화시킨다. 이런 초기화가 필요한 이유는 에러 코어 내부의 어떤 부분에 에러가 발생하였는지 알 수 없기 때문에 코어 초기화를 통해 에러 코어가 안정화된 상태로 전환하기 위해서다.

2. 멀티플렉서를 활용한 인스트럭션 버스의 직접 연결

시스템의 모든 코어는 초기화 과정에서 리셋 벡터 데이터를 인스트럭션 버스를 통해 직접 수신해야 한다. 이

Fig. 4. Waveform of error occurrence and recovery process in CORE1.
그림 4. CORE1에서의 에러 발생 및 복구 과정 파형

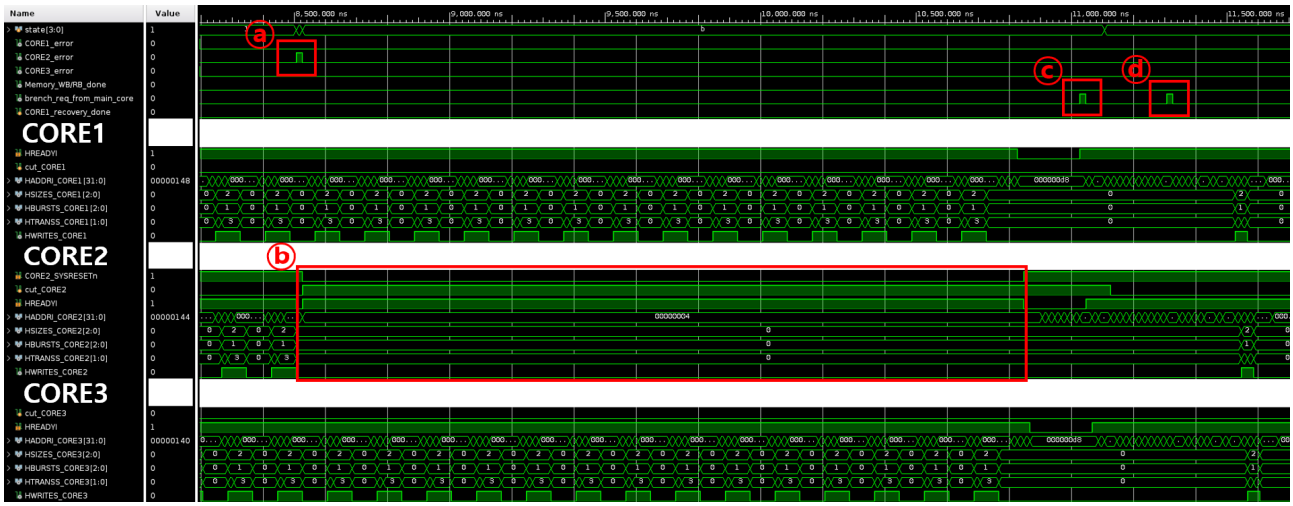

Fig. 5. Waveform of error occurrence and recovery process in CORE2.
그림 5. CORE2에서의 에러 발생 및 복구 과정 파형

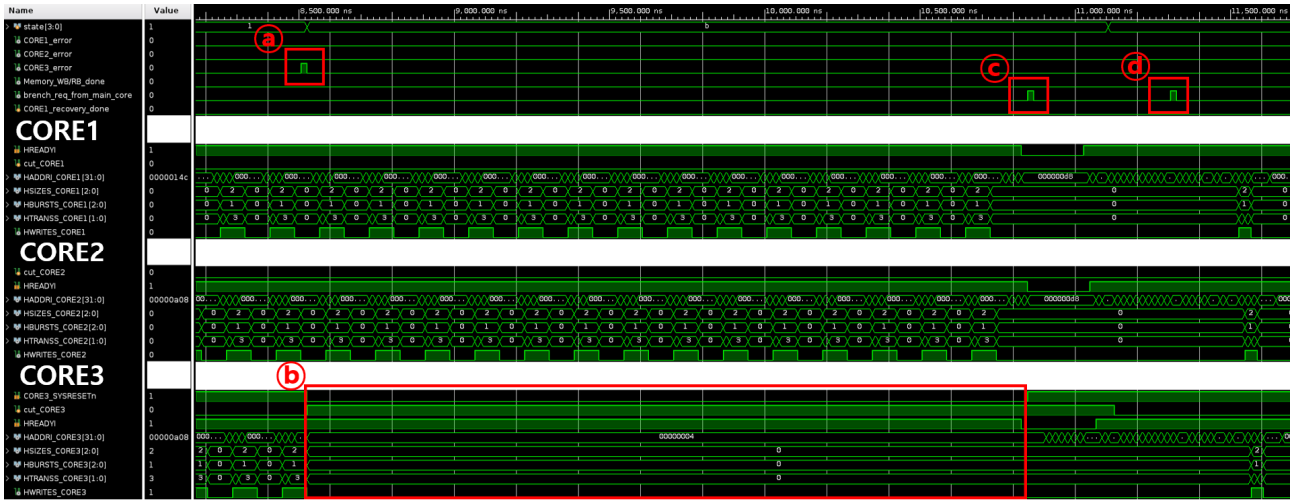

Fig. 6. Waveform of error occurrence and recovery process in CORE3.
그림 6. CORE3에서의 에러 발생 및 복구 과정 파형

를 위해 초기화 중인 에러 코어의 인스트럭션 버스를 시스템의 인스트럭션 버스에 직접 연결하였다. 이 연결은 멀티플렉서를 통해 수행되며, 에러 코어의 초기화가 완료된 후에는 D-TCLS 코어 동기를 유지하기 위해 에러 코어가 다시 적절한 지연이 적용된 인스트럭션 데이터를 받을 수 있도록 설정한다.

3. 정상 코어의 GPR, SPR값 복사 및 "BX LR" 명령어 패치

초기화 과정이 완료된 에러 코어는 시스템 코어의 프로그램 카운터(PC) 값과 동기화해야 한다. 이를 위해 첫 단계로, 정상 코어의 일반 목적 레지스터(GPR) 값 및 특수 목적 레지스터(SPR) 값을 초기화된 에러 코어로 복사한다. 복사 작업이 완료되면, 복사된 GPR 및 SPR 값들을 활용해 시스템 코어와 동기화된 상태로 동작하게 된다.

다음 단계에서는, Thumb 명령어 셋의 'BX LR' 명령어를 코어에 패치한다. 이 명령어를 패치하기 위해 코어 제어기 내부의 명령어 입력 기능을 이용해 해당 코어의 인스트럭션 데이터 입력으로 'BX LR' 명령어를 출력하도록 한다. 이를 통해 초기화된 에러 코어는 시스템 코어의 PC 값과 동기화된 상태로 제대로 동작하게 된다.

4. 시스템 버스의 정지 시점 조절

복구 과정이 완료된 에러 코어는 D-TCLS 모드를 원활하게 수행하기 위해 적절한 타이밍에 동작을 시작해야 한다. 이를 위해 각 코어들의 정지(stall) 상태를 관리하는 AHB 버스의 HREADY 신호를 코어 제어기에서 조정하여, D-TCLS가 원활히 동작할 수 있도록 시스템 동기화를 최적화한다.

Ⅳ. 시뮬레이션 및 검증

시뮬레이션 환경에서는 소프트 에러를 모의하기 위해 D-TCLS 시스템에 일시적인 에러 주입(Error Injection) 기능을 Verilog HDL로 설계한 블록을 추가하였다. 이를 통해 로직의 출력값, 플립플롭 및 레지스터의 출력값, 플립플롭 및 레지스터의 저장값에 인위적으로 에러를 주입하고 시뮬레이션을 통한 검증을 진행하였다.

1. CORE1에서 에러 발생 및 복구

그림 4는 CORE1에서 에러 발생 및 복구 과정을 보여준다. CORE1에서 에러가 발생한 후, 메모리 Roll-Back과 Write-Back 과정을 거친 뒤 메인 코어를 CORE2로 변경한 후, D-DCLS 모드로 전환하여 시스템을 동작시킨다. 메인 코어의 Branch 타이밍에서 CORE1의 복구를 진행하며, 메인 코어를 CORE2에서 다시 CORE1로 전환한 후 시스템이 정상적으로 동작한다.

그림 4의 ⓐ는 CORE1에서 에러가 발생하였을 때 코어 제어기가 해당 에러를 감지한 상태를 나타낸다. 이어서 메모리 RB/WB 과정을 거친 후, 메인 코어를 CORE2로 변경한 것을 그림 4의 ⓑ에서 확인할 수 있다. 그림 4의 ⓒ는 메인 코어가 Branch 명령어를 발행할 때까지 D-DCLS 모드에서 시스템이 작동하는 모습을 보여준다. 이때, 에러가 발생한 CORE1는 대기 상태로 전환된다. 그림 4의 ⓓ는 메인 코어에서 Branch 명령어를 발행하면서 코어 제어기가 CORE1의 복구 과정을 시작하는 시점을 나타낸다. 그림 4의 ⓔ는 CORE1의 복구가 완료되어, 시스템의 메인 코어를 CORE1로 다시 전환하는 시점을 보여주며, 이것은 CORE1의 복구가 완료됐고, D-TCLS 모드로 시스템이 동작한다는 것을 나타낸다.

2. 서브 코어에서의 에러 발생 및 복구

그림 5와 그림 6은 각각 서브 코어인 CORE2와 CORE3에서 에러 발생 및 복구 과정을 보여준다. 해당 코어를 일시 정지시키며 시스템을 D-DCLS 모드로 전환하고 메인 코어에서 Branch 명령어 발행 타이밍에 복구를 시작하여 최종적으로 시스템을 D-TCLS 모드로 재전환한다.

그림 5, 6의 ⓐ는 코어 제어기가 각 코어에서 발생한 에러를 감지한 상태를 보여주는 신호이다. 그림 5, 6의 ⓑ에서는 에러가 발생한 서브 코어가 일시 정지되는 것을 나타내는 동시에 다른 코어들은 정상적으로 작동하고 있음을 알 수 있다. 그림 5, 6의 ⓒ는 메인 코어가 Branch 명령어를 발행하면서, 코어 제어기가 에러가 발생한 서브 코어의 복구 과정을 시작하는 시점을 나타낸다. 그림 5, 6의 ⓓ는 에러가 발생한 서브 코어의 복구가 완료되고, 시스템이 다시 D-TCLS 모드로 재전환하여 에러가 발생한 코어가 다시 투표 역할을 수행하기 시작하는 것을 나타낸다.

Ⅴ. 결론

본 논문에서는 삼중 지연 락스텝 코어 시스템에서 개별 코어에 에러가 발생한 경우 이를 복구하는 방법을 제시하였다. 코어 초기화, 시스템과 코어 간의 인스트럭션 버스 연결, 그리고 필요한 정보 및 명령어의 복사와 패치

를 통해 에러 코어를 복구할 수 있음을 보여주었다. 이를 위해 초기화 절차를 정의하고, MUX를 사용하여 시스템과 코어 간의 인스트럭션 버스를 직접 연결하였다. 또한, Thumb 명령어 셋의 'BX LR' 명령어를 사용하여 프로그램 카운터(PC)를 정상 코어의 PC 값으로 복사하고, 이를 에러 코어에 덮어 쓰는 방법을 제시하였다.

특히, 코어 복구 절차를 시스템 메인 코어의 내부 파이프라이닝이 플러시 처리되는 Branch 타이밍에 진행하도록 설정함으로써, 코어 내부의 다른 신호들을 전부 복사할 필요 없이 GPR 값만 복사할 수 있게 하여 시스템의 복잡도를 크게 줄였다.

본 논문에서 제안한 방법을 통해 에러가 발생한 코어의 초기화와 복구를 수행하면, 에러가 발생했을 때 전체 시스템의 초기화를 하지 않고 해당 코어를 복구하여 시스템의 안정적이고 계속적인 동작을 보장할 수 있다.

References

[1] ISO 26262-1 : 2018, "Road Vehicle – Functional Safety – Part 1 : Vocabulary," https : //www.iso.org /standard/68383.html

[2] S. Choi, "Establishment of Extended Model for Determining and Evaluating ASIL in the ISO 26262 Automotive Functional Safety System," *Journal of the Korean Institute of Plant Engineering,* vol.22, no.2, pp.39-56, 2017.

[3] W. Lee, K. We, S. Kim and C. Lee, "Simulator Structure for Lockstep ECU," *Proceedings of Korea Computer Congress,* pp.1508-1510, 2017.

[4] S. Yang, J. Choi, and S. Lee, "Design of Delayed Triple-Core Lock-Step Processor with Memory Rollback for Automotive Applications," *J.inst.Korean.electr.elctron.eng.,* vol.26, no.4, pp. 628-632, 2022. DOI: 10.7471/ikeee.2022.26.4.628

[5] S. Yang, "Design of a Triple-core, Delay-Locked Loop System with Memory Rollback Capability Using ARM Cortex Cores," Master Thesis, Soongsil University, 2023.

[6] ARM, "Cortex-M3 Devices Generic User Guide," https : //developer.arm.com/ documentation/dui0 552/a/?lang=en

BIOGRAPHY

Seonghyun Yang (Member)

2021 : BS degree in Electronic Engineering, Soongsil University.
2023 : MS degree in Electronic Engineering, Soongsil University.
2023~ : Candidate for Ph.D degree in Electronic Engineering, Soongsil University.
〈Main Interest〉 Automotive SoC, Processor SoC, AI SoC

Juho Kim (Member)

2022 : BS degree in Electronic Engineering, Soongsil University.
2022~ : Candidate for MS degree in Electronic Engineering, Soongsil University.
〈Main Interest〉 Automotive SoC, Processor SoC, AI SoC

Seongsoo Lee (Life Member)

1991 : BS degree in Electronic Engineering, Seoul National University.
1993 : MS degree in Electronic Engineering, Seoul National University.
1998 : PhD degree in Electrical Engineering, Seoul National University.
1998~2000 : Research Associate, University of Tokyo
2000~2002 : Research Professor, Ewha Womans University
2002~Now : Professor in School of Electronic Engineering, Soongsil University
〈Main Interest〉 AI SoC, Automotive SoC, Security SoC, Processor SoC, Power Management SoC, Battery Management SoC, Reliability and Safety