J. Korean Soc. Aeronaut. Space Sci. 48(12), 1005-1012(2020)
DOI:https://doi.org/10.5139/JKSAS.2020.48.12.1005
ISSN 1225-1348(print), 2287-6871(online)

항공전자 시스템을 위한 PCI-Express 버스의 결함감내 구조

김성준[1], 김경훈[2], 전용기[3]

A Fault-Tolerant Architecture of PCI-Express Bus for Avionics Systems

Sung-Jun Kim[1], Kyong-Hoon Kim[2] and Yong-Kee Jun[3]
Korea Aerospace Industries, LTD[1]
Kyungpook National University[2], Gyeongsang National University[3]

ABSTRACT

Avionics systems that use the PCI-Express bus unfortunately cannot use at least one I/O device if the bus fails, because the I/O device is connected to CPU through only one PCI-Express channel. This paper presents a fault-tolerant architecture of the PCI-Express bus for avionics systems, which tolerates one channel failure with help of the other redundant channel that has not been failed. In this architecture, each redundant PCI-Express channel connects a corresponding port of CPU to each switch logic of channels to provide each I/O device through a switched fault-tolerant channel. This paper includes the results of experimentation to show that the architecture detects the faulty condition in real time and switches the channel to the other redundant channel which has not been failed, when the architecture meets a failure.

초 록

PCI-Express 버스를 적용하는 항공전자 시스템은 CPU와 입출력 장치를 하나의 채널만을 사용하여 연결하기 때문에, 불행하게도 그 채널에 고장이 발생하면 적어도 하나의 입출력 장치를 사용할 수 없게 되는 문제가 있다. 본 논문은 항공전자 시스템을 위한 PCI-Express 버스의 결함감내 구조를 제시하기 위해서, PCI-Express 채널을 이중화하여 하나의 채널에 고장이 발생하여도 고장이 발생하지 않은 다른 채널을 통해 여전히 정상적으로 기능하는 버스 구조를 제시한다. 논문에서 제시하는 버스 구조는 두 개의 CPU port에서 출력된 이중적 PCI-Express 버스 신호를 각각의 switch 회로에 입력되게 하고, 이 회로가 각 입출력 장치에 결함을 감내하도록 선택된 독립된 버스 채널을 제공하게 한다. 본 논문에서는 제시하는 버스 구조를 구현 및 실험하여 하나의 PCI-Express 버스에 고장이 발생하면, 그 고장 상황을 실시간으로 감지되고, 고장이 발생하지 않은 다른 버스로 채널을 전환되어 정상적으로 통신이 수행되는 것을 보인다.

Key Words: Avionics System(항공전자 시스템), PCI-Express Bus(PCI-Express 버스), Fault-tolerant Bus(결함감내 버스), Bus Architecture(버스 구조)

Ⅰ. 서 론

오늘날의 항공기는 항공전자 계통과 연동되는 모든 항공기 계통들을 활용, 제어 및 관리하기 위해서 항공기 운용에서 요구되는 기능 및 품질을 제공하는 임베디드 컴퓨터 시스템이 중요하다. 일반적으로 항

† Received: January 8, 2020 Revised: October 13, 2020 Accepted: October 20, 2020
[1] Senior Research Engineer, [2,3] Professor
[3] Corresponding author, E-mail: jun@gnu.ac.kr, ORCID 0000-0002-4753-3651

공전자 계통에서는 소형화, 경량화, 고신뢰성 등이 우선적으로 요구되기 때문에, 이러한 임베디드 컴퓨터 시스템은 각 부분체계에서 공통적으로 필요한 단위 기능을 모듈화하여 공유하는 통합형 항공전자 시스템으로 발전되어 왔다[1].

이러한 임베디드 컴퓨터 시스템에서는 CPU 모듈과 입출력 장치 사이에 고속으로 데이터를 전달하기 위해서 고속의 직렬 버스인 PCI-Express 버스[2]를 사용하는 사례[9-15]가 많아지고 있다. PCI-Express 버스는 기존의 병렬버스와 비교하여 많은 장점을 제공한다. 먼저 저전압 차동 신호를 사용함으로써 노이즈가 적게 발생한다. 그리고 데이터를 직렬 방식으로 통신하기 때문에 수십 개의 주소/데이터 버스 대신에 매우 적은 수의 신호선으로 고속의 통신이 가능하다. 뿐만 아니라 multi-drop 방식이 아닌 point-to- point 통신방식을 적용하기 때문에 하나의 모듈에서 발생한 버스의 고장이 다른 모듈의 통신에 영향을 주지 않는다.

그러나 PCI-Express 버스도 버스 자체에서 고장이 발생하면 고장의 부위에 따라 전체 모듈 또는 일부의 모듈이 사용될 수 없는 상태에 빠지게 된다. 이러한 문제는 PCI-Express 버스가 내재하고 있는 결함으로 인해서 고장이 발생해도 즉시 그 고장을 탐지하고 회피하는 결함감내(fault-tolerant) 구조로써 해결될 수 있다. 본 논문은 PCI-Express 버스의 결함감내 구조를 제공하기 위한 핵심적인 모듈로서 CPU에서 이중화된 버스와 입출력 장치의 local bus를 연결하는 인터페이스를 제시한다.

본 논문에서 제시하는 결함감내 PCI-Express 버스는 독립된 PCI-Express switch 회로를 중복적으로 내장하여 각각의 입출력 장치에 버스의 결함을 감내하는 PCI-Express 채널을 제공한다. 입출력 장치를 위한 모듈은 switch 회로가 제공하는 중복된 채널을 FPGA[3] 인터페이스를 이용하여 통합한다. 사용된 FPGA는 일반적인 PCI-Express Switch 대비 낮은 failure rate[4] 갖는 고신뢰성을 제공할 뿐만 아니라, 입출력 모듈을 소형화하고 개발 비용이 절감될 수 있게 한다.

본 논문의 본론인 II절에서는, 항공전자 시스템에서 사용하는 PCI-Express 버스 구조와 그 신뢰성의 관점에서의 문제점을 소개하고, 이러한 문제를 해결하기 위한 결함감내 버스 구조를 제시하고, 제시된 버스 구조를 대상으로 한 신뢰도 분석 결과를 보이고, 마지막으로 구현된 버스 구조의 실험 결과를 제시한다. III절에서는 본 논문의 결론을 보인다.

II. 연구 배경

본 논문의 연구 분야는 통합형 항공전자 시스템에서 사용하는 PCI-Express 버스 분야이다. 통합형 항공전자 시스템[1]은 각 부분체계에서 공통적으로 필요한 단위 기능을 모듈화하여 공유할 수 있게 하는 컴퓨터 시스템들과 이들을 연결하는 MIL-STD-1553B [5] 또는 ARINC-429 표준 등의 통신망으로 구성된다. 그리고 임베디드 컴퓨터 시스템 내부에서는 CPU 모듈과 입출력 또는 데이터 처리 모듈 간의 통신을 위해서 병렬 버스와 직렬 버스를 사용한다.

임베디드 컴퓨터 시스템의 병렬 버스는 여러 개의 데이터 신호선 및 어드레스 신호선을 사용하여 데이터를 동시에 전달하는 방식으로써 Local Bus, VME (VERSA Module Eurocard) 버스[6], PCI 버스[7] 방식이 있다. 하지만 병렬 버스는 다음과 같은 이유로 인해서 항공전자 시스템에 적용하기 어렵다. 첫째, 병렬 버스는 전자파 노이즈가 발생하는 구조이다. 이 유형의 버스는 100 MHz 미만의 대역폭을 가는데, EMI(Electro Magnetic Interference) 및 노이즈 효과는 대역폭이 높을수록 나빠진다. 둘째, 여러 개의 회선을 공통으로 사용하기 때문에 각 회선에 연결된 회로에 문제가 발생하면 해당 데이터의 전송에 오류를 유발한다. 그러므로 버스의 고장 확률을 높이게 된다. 세 번째, 데이터를 읽거나 쓰기 위해서는 버스를 선점해야 한다. 그러므로 다른 장치는 버스의 사용이 끝날 때까지 기다려야 하는 병목 현상이 발생하게 된다. 그 외에도, 어드레스 버스, 데이터 버스, 제어 버스와 같이 수많은 신호들이 연결되기 때문에 PCB(Printed Circuit Board) 구조와 커넥터 구조를 정하기 어려운 문제가 있다.

본 논문에서 다루는 PCI-Express 버스[2]는 2002년 PCI 표준으로 발표된 직렬 버스로서 높은 시스템 버스 대역폭, 작은 수의 입출력 핀 및 그에 따른 적은 물리적 면적, 그리고 뛰어난 확장성을 가진 버스이다. 그리고 PCI-Express 버스는 고속, 저잡음, 저전력 방식인 LVDS(Low Voltage Differential Signaling) 방식[8]으로 통신을 한다. LVDS 방식은 잡음 내성 및 잡음 마진을 증가시키고 신호 간섭(cross-talk)에 의한 신호의 왜곡에 강하기 때문에 EMI 노이즈를 자체적으로 줄일 수 있다. 그리고 버스 간 길이의 차이에 의한 전송 신호의 타이밍 및 위상 오류(skew)가 발생하지 않는다.

하지만 PCI-Express 버스를 적용하는 항공전자 시스템은 CPU와 입출력 장치를 하나의 채널만을 사용하여 연결하기 때문에, 불행하게도 그 채널에 고장이 발생하면 적어도 하나의 입출력 장치를 사용할 수 없게 되는 문제가 있다. 본 절에서는 항공전자 시스템에서 PCI-Express 버스를 사용하는 기존의 관련 연구를 신뢰성의 관점에서 문제점을 제기한다.

2.1 관련 연구

PCI-Express 버스를 사용하려면 버스 회선을 연결하는 bridge 회로가 필요한데, CPU의 메모리 버스를

연결하기 위한 host bridge, 장치를 연결하기 위한 local bridge 등이 있다. PCI-Express 버스를 적용하는 항공전자 시스템은 이러한 bridge 회로를 별도로 설계한 경우[9]도 있지만, 더 높은 신뢰성을 위해서 CPU나 전용 controller에 내장한 경우[10,11]도 있으며, CPU와 연결되는 장치의 수를 늘리기 위해서 bridge 회로에 switch 회로를 추가한 경우[12,13]도 있다.

첫 번째의 bridge 회로를 별도로 설계하는 경우[9]는 PCI-Express bus 회선에 host bridge와 local bridge를 추가하고 그 출력을 주변장치 연결하였다(Fig. 1). 두 번째는 bridge 회로가 CPU 혹은 전용 controller에 내장되는 경우[10,11]로서, CPU에서 출력된 버스 신호를 다른 모듈의 CPU(Fig. 2) 또는 전용 controller(Fig. 3)에 직접적으로 연결하는 경우이다. 나머지는 bridge 회로에 switch 회로를 추가하는 경우[12-15]로서, host bridge가 내장된 CPU에 PCI-Express switch 회로를 연결하여 채널수를 늘리고 난 후에, PCI-Express local bridge를 내장한 입출력 모듈에 연결하였다(Fig. 4).

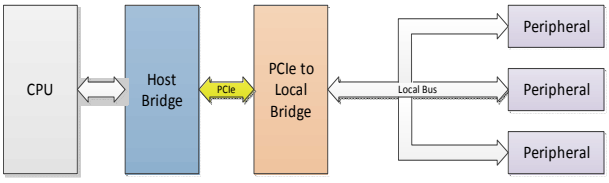

Fig. 1. separate host bridge and local bridge

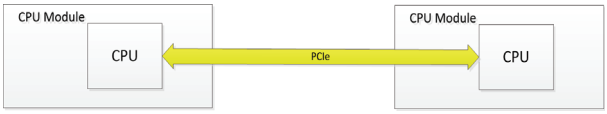

Fig. 2. host bridges embedded in CPUs

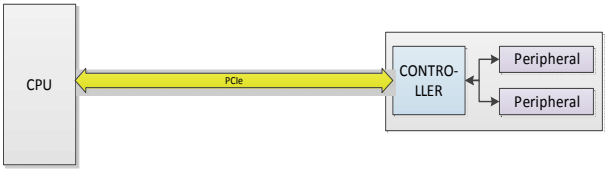

Fig. 3. a local bridge embedded in a controller

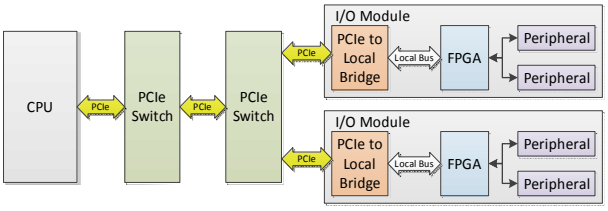

Fig. 4. embedded bridges and switches

2.2 문제 제기

PCI-Express 버스를 적용하는 항공전자 시스템[9-13]은 CPU와 입출력 장치를 하나의 채널만을 사용하여 연결하기 때문에, 불행하게도 그 채널에 고장이 발생하면 적어도 하나의 입출력 장치를 사용할 수 없게 되는 문제가 있다.

먼저, bridge 회로를 별도로 설계하는 경우[9]를 보자. 이 경우에는 host bridge 혹은 local bridge와 그 연결 부위에 고장이 발생하게 되면 곧 PCI-Express bus의 고장이 되는 문제점이 있다. 두 번째로, bridge 회로가 CPU 혹은 전용 controller에 내장되는 경우[14,15]를 보자. 이 경우에는 CPU 또는 controller의 고장을 배제한다고 하더라도 그 연결 부위에 고장이 발생하게 되면 곧 PCI-Express bus의 고장이 되는 문제점이 있다. 나머지 사례로, bridge 회로에 switch 회로를 추가하는 경우[12-15]를 보자. 이 경우에서 발생되는 고장은 부분적인 고장과 전반적인 고장으로 나눌 수 있다. 먼저, local bridge 또는 switch 회로와의 연결 부위가 고장이 난 상황에서는 관련되는 입출력 모듈만 사용할 수 없는 부분적인 고장이 된다. 하지만 CPU와 switch 회로의 연결 부위에 고장이 발생되거나 switch 회로에 고장이 발생하게 되면 PCI-Express bus의 전반적인 고장이 되는 문제점이 있다.

이러한 문제는 PCI-Express 버스가 내재하고 있는 결함으로 인해서 고장이 발생해도 즉시 그 고장을 탐지하고 회피하여 내재된 결함(fault)을 감내하는(tolerant) 버스 인터페이스로써 해결될 수 있다.

III. 결함감내 PCI-Express Bus

PCI-Express 버스의 결함감내 구조를 제공하기 위해서, 본 절은 CPU에서 이중화된 PCI-Express 채널과 입출력 모듈의 local bus 사이를 연결하는 인터페이스 회로의 구조를 제시한다.

3.1 규격

본 논문이 제시하는 인터페이스는 CPU의 이중화된 PCI-Express 채널[14,15] 중에서 PCI-Express 버스의 결함을 감내하는 채널을 선택하여 입출력 모듈의 local bus와 통신한다. 이 인터페이스는 PCI-Express 버스의 device driver 소프트웨어가 구동하는데, 중복된 채널 중에서 고장이 발생한 채널을 탐지하고, 다른 채널을 해당되는 입출력 모듈의 local bus에 연결한다. Fig. 5는 본 논문에서 제시된 device driver가 수행하는 task schedule을 보인다.

결함감내 PCI-Express 버스의 device driver는 버스를 통해서 입출력 모듈의 건전성(health)을 검사하는데, 대부분의 항공전자 시스템이 사용하는 50ms

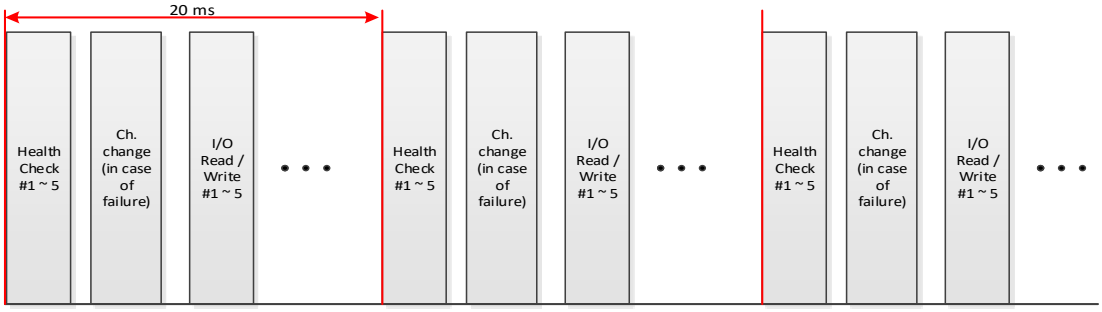

Fig. 5. task scheduling of the device driver

(20Hz) 주기[16]로 수행한다. 그리고 PCI-Express 채널의 고장이 탐지되면, 인터럽트 핸들러가 해당 입출력 모듈의 PCI-Express 채널을 고장이 발생하지 않은 다른 CPU 포트의 채널로 변경하여 통신을 재개한다.

3.2 설계

PCI-Express 버스의 결함을 감내하기 위해서 본 논문에서 제시하는 결함감내 인터페이스는 Fig. 6에서 보이는 바와 같이 FPBI (Fault-Tolerant PCI-Express Bus Interface), 2개의 내부 PCI-Express 채널, FPGA로 구현된 FLBI (Fault-Tolerant Local Bus Interface) 등의 3개의 요소로 구성된다.

FPBI는 CPU에서 이중화된 PCI-Express 채널 중에서 버스의 결함을 감내하는 채널을 선택하는 인터페이스이다. 그리고 FPBI가 제공하는 중복된 채널은 입출력 모듈을 위해서 통합하는 인터페이스인 FLBI에 연결되는데, Microsemi社의 SmartFusion2 FPGA[3]에 내장된 PCI-Express 인터페이스를 이용하여 구현된다. 여기서 CPU는 Port-A 혹은 Port-B에 연결된 버스 채널을 통해서 FPGA의 Port-A 혹은 Port-B에 접근할 수 있다.

FLBI는 입출력 모듈에 내장되는데, 독립된 두 개의 PCI-Express 채널을 통해 들어온 명령 또는 데이터가 어느 포트에 연관된 것인지 식별하기 위해서, ORing Logic을 AMBA(Advanced Microcontroller Bus Architecture)[17]를 이용하여 FPGA로 제공한다. 쓰기 명령일 경우에는 이를 내부의 메모리 또는 레지스터에 전달한다. 반대로 읽기 명령일 경우에는 메모리 또는 레지스터에서 출력된 데이터를 요청한 포트로 전달한다.

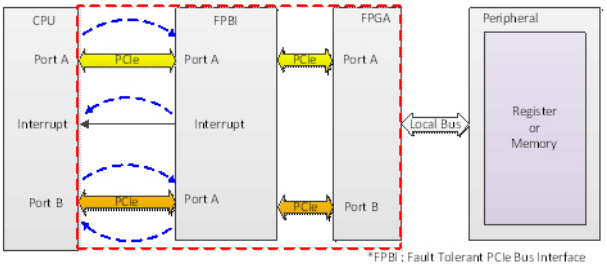

Fig. 6. the fault-tolerant PCI-Express bus

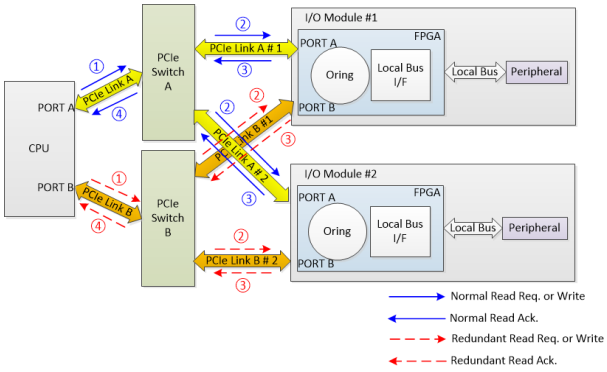

Fig. 7. the design of FPBI

FPBI는 PCI-Express 채널 중에서 버스의 결함을 감내하는 채널을 선택하기 위해서, 독립된 switch 회로를 두 개씩 중복적으로 내장하여 각 출력에 중복된 PCI-Express 채널을 제공한다. FPBI의 구조는 Fig. 7에서 보인다. 그림에서 보는 바와 같이, 두 개의 CPU 포트는 각각 대응되는 두 개의 switch 회로와 두 개의 통신 경로(link)를 구성하고 있다. 그리고 각 switch 회로는 두 개의 FPGA-FLBI 포트와 두 개의 통신경로를 구성하고 있다. 이러한 통신경로로 구성되는 버스의 통신 채널은 device driver가 관리하는데, 다음과 같은 두 가지 작업이 있다.

- 통신경로의 상태를 등록(Fig. 8)
- 통신경로의 고장탐지 및 경로변경(Fig. 9)

통신 경로의 상태를 등록하는 작업(Fig. 8)은 입출력 모듈을 검사하기 위해서 두 CPU 포트에 연결된 통신 경로를 통해서 정해진 입출력 모듈의 식별자(identifier, ID)를 확인하고 그 성공 여부에 따라서 통신 경로의 상태를 등록한다. 모든 통신 경로가 유효하면, Fig. 7에서 보인 바와 같이 CPU Port-A와 FPGA Port-A를 연결하는 경로를 기본적인 채널(normal channel)로 등록한다. 그러나 유효하지 않는 통신 경로가 확인되면, 그 경로가 결함이 있음을 등록하고 연결할 수 없는 상태로 등록한다.

채널의 고장을 탐지하고 통신경로를 변경하는 작업(Fig. 9)은 고장이 발생된 통신경로가 아닌 다른 통신경로의 상태에 의존적이다. 다른 통로도 마찬가지로 결함이 있는 경우는 PCI-Express 버스가 고장인

상태가 되지만, 그렇지 않는 경우에는 고장인 통신경로를 결함 상태로 등록하고, 연결하는 통로를 다른 통로로 변경한다.

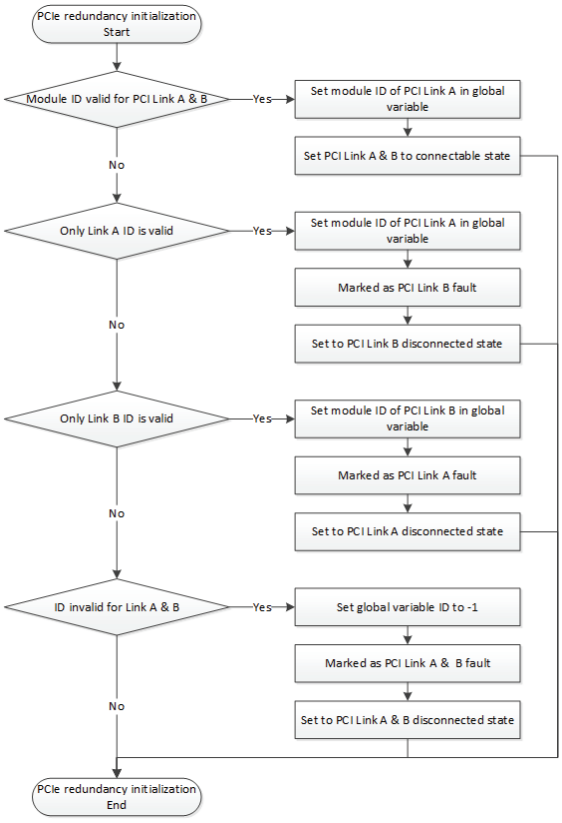

Fig. 8. registering the states of links

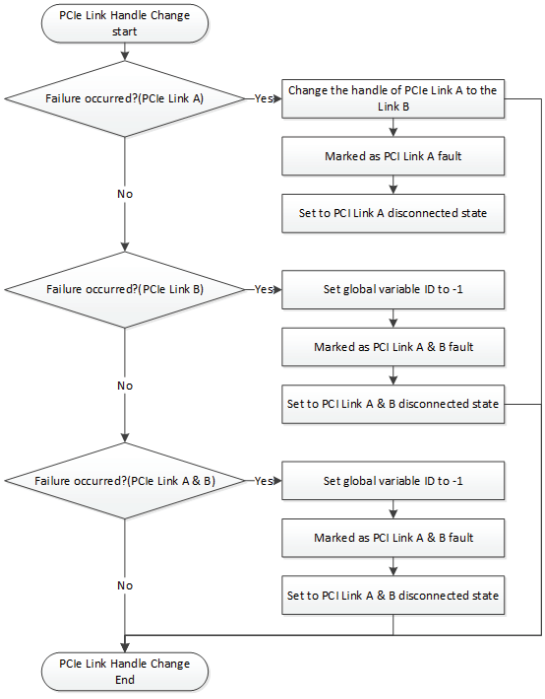

Fig. 9. changing PCI-Express link handles

3.3 동작

컴퓨터 시스템이 초기화될 때, 운영체제는 CPU에 연결된 다음의 PCI-Express 요소들을 인식하고 고유한 주소를 부여할 수 있다. Table 1은 각 I/O module의 Port 별로 부여된 주소의 예를 보인다.

- PCI-Express switch A,
- PCI-Express switch B,
- I/O module #1의 FPGA Port-A/B,
- I/O module #2의 FPGA Port-A/B

본 논문에서 제시하는 고장감내 PCI-Express 버스가 고장이 발생하지 않은 정상적인 상황을 생각해 보자. 입출력 모듈 #1의 module ID를 읽는 경우에는, (base address + module ID) = (0x8000_0000 + 0x100) = (0x8000_0100) 번지에 읽기 작업을 시도한다. 그리고 입출력 모듈 #2의 module ID를 읽기 위해서는 모듈 #1의 경우와 마찬가지 방법으로 (0x8100_0100) 번지를 base address로 하여 읽기 작업을 시도한다. 이와 같이 CPU는 Port-A를 통해서 switch-A와 각 입출력 모듈의 Port-A를 통해 입출력 장치에 접근할 수 있게 된다.

다음으로, 본 논문에서 제시하는 PCI-Express 고장감내 버스의 switch와 module 포트에 고장이 발생한 상황을 생각해 보자. 첫째로, switch에 고장이 발생한 경우(Fig. 10)에는, 먼저 모듈 #1의 module ID를 읽기 위해 Port-A의 (0x8000_0100) 번지로 읽기를 시도하면 오류가 발생한다. 그러면 다음에는 Port-B의 (0xC000_0100) 번지로 읽기를 시도하고, 마찬가지로 오류가 발생한다. 입출력 모듈 #2의 module ID를 읽는 경우에도 (0x8100_0100) 번지와 (0xC100_0100) 번지에 읽기를 시도하고 오류가 발생한다.

Table 1. addresses mapped to I/O modules

Target	Address Range
Module #1 - Port-A	0x8000_0000-0x800F_FFFF
Module #2 - Port-A	0x8100_0000-0x810F_FFFF
Module #1 - Port-B	0xC000_0000-0xC00F_FFFF
Module #2 - Port-B	0xC100_0000-0xC10F_FFFF

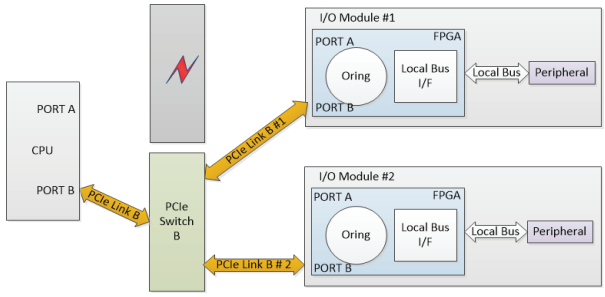

Fig. 10. channels in a switch-A failure

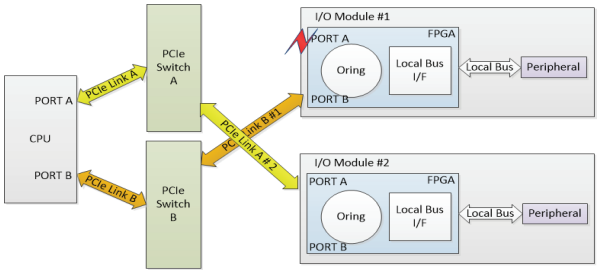

Fig. 11. channels in an FPGA Port-A failure

다음으로, 입출력 모듈 #1의 FPGA 포트에 고장이 발생한 경우(Fig. 11)를 생각해 보자. 모듈 #1의 module ID를 읽기 위해 (0x8000_0100) 번지로 읽기를 시도하고 오류가 발생한다. 그러면 다음에는 (0xC000_0100) 번지로 읽기를 시도하고 마찬가지로 오류가 발생한다. 하지만 고장이 발생하지 않은 모듈 #2의 경우에는 (0x8100_0100) 번지를 통해 읽기를 성공하게 된다.

IV. 분석 및 실험

4.1 신뢰도 분석

본 논문에서 제시하는 PCI-Express 버스 시스템의 신뢰도를 분석하기 위해서 결함율 모델에 따른 수치적 분석법을 적용한다. 한 시스템의 신뢰도 함수 $R(t)$는 시작 시각 0에서부터 작동하고 있을 때, 이 시스템이 시각 t에서도 여전히 작동하고 있을 확률이다. 만일 한 시스템의 결함율이 λ로 주어진다면, 그 시스템의 신뢰도 함수 $R(t)$는 $e^{-\lambda t}$ 함수를 따른다 ($t > 0$).

먼저, 관련 연구에서 제시한 이전의 버스 시스템(Fig. 4)의 신뢰도 함수를 유도하기 위해서, CPU에서 주변장치 하나에 도달하는 경로를 기준으로 구성요소들을 도식화하면 Fig. 12(a)와 같다. 즉, 여러 개의 컴포넌트가 직렬로 연결되어 있으며, 직렬로 연결된 시스템의 신뢰도 함수는 각 컴포넌트의 신뢰도 함수의 곱으로 정의된다. 그러므로 이전 시스템의 신뢰도 함수 $R_{prev}(t)$를 Table 2에서 정한 표현으로 정의하면 수식 (1)과 같다.

$$\begin{aligned} R_{prev}(t) &= R_{PCB}(t) \cdot R_{Switch}(t) \cdot R_{PCB}(t) \cdot R_{Bridge}(t) \cdot R_{PCB}(t) \cdot R_{FPGA}(t) \\ &= e^{-\lambda_{PCB}t} \cdot e^{-\lambda_{Switch}t} \cdot e^{-\lambda_{PCB}t} \cdot e^{-\lambda_{Bridge}t} \cdot e^{-\lambda_{PCB}t} \cdot e^{-\lambda_{PCB}t} \cdot e^{-\lambda_{FPGA}t} \\ &= e^{-(\lambda_{PCB}+\lambda_{Switch}+\lambda_{PCB}+\lambda_{Bridge}+\lambda_{PCB}+\lambda_{FPGA})t} \end{aligned} \quad (1)$$

본 논문에서 제시하는 버스 시스템은 CPU에서 주변장치 하나에 도달하는 경로를 기준으로 구성요소들을 도식화하면 Fig. 12(b)와 같다. 즉, CPU로부터 FLBI-FPGA까지 두 개의 동일한 컴포넌트들이 병렬로 연결된다. 이 시스템의 신뢰도 함수를 유도하기 위해서 먼저 (PCB, Switch, PCB) 컴포넌트로 구성된 부분 시스템을 PSP라고 하자. 그러면 이 부분 시스템의 신뢰도 함수는 구성요소들이 직렬로 연결되어 있으므로 수식 (2)와 같이 정의된다.

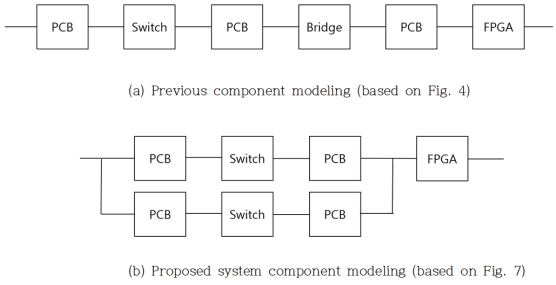

Fig. 12. component models for analysis

Table 2. notations for reliability analysis

Component	Reliability Function	Defect Rate	Measured Value
PCIe, local bus	$R_{PCB}(t)$	λ_{PCB}	0.001360
PCIe switch	$R_{Switch}(t)$	λ_{Switch}	0.070052
PCIe to local bridge	$R_{Bridge}(t)$	λ_{Bridge}	0.033
FPGA	$R_{FPGA}(t)$	λ_{FPGA}	0.026527

$$\begin{aligned} R_{PSP}(t) &= R_{PCB}(t) \cdot R_{Switch}(t) \cdot R_{PCB}(t) \\ &= e^{-\lambda_{PCB}t} \cdot e^{-\lambda_{Switch}t} \cdot e^{-\lambda_{PCB}t} \\ &= e^{-(\lambda_{PCB}+\lambda_{Switch}+\lambda_{PCB})t} \end{aligned} \quad (2)$$

그리고 본 논문에서 제시하는 시스템은 두 개의 PSP가 병렬로 연결되고, 그 이후에 FPGA 컴포넌트가 직렬로 연결된다. 만일 컴포넌트 (A, B)가 병렬로 구성된 시스템이 있다고 하면, 이 시스템의 신뢰도 함수는 $[1-(1-R_A(t))(1-R_B(t))]$으로 정의된다. 그러므로 본 논문에서 제시하는 시스템의 신뢰도 함수는 수식 (3)과 같이 유도할 수 있다.

$$\begin{aligned} R_{proposed}(t) &= 1-(1-R_{PSP}(t)) \cdot (1-R_{PSP}(t)) \cdot R_{FPGA}(t) \\ &= 1-(1-e^{-\lambda_{PSP}t})(1-e^{-\lambda_{PSP}t}) \cdot e^{-\lambda_{FPGA}t} \\ &= (2e^{-\lambda_{PSP}t} - e^{-2\lambda_{PSP}t}) \cdot e^{-\lambda_{FPGA}t} \\ &= e^{-(\lambda_{PSP}+\lambda_{Switch}+\lambda_{PSP}+\lambda_{FPGA})t}(2-e^{-(\lambda_{PSP}+\lambda_{Switch}+\lambda_{PSP})t}) \end{aligned} \quad (3)$$

수식 (1)과 수식 (3)에서 유도한 신뢰도 함수를 기반으로 하여, 이전의 시스템과 본 논문에서 제시된 시스템의 신뢰도를 분석하면 Fig. 13과 같다. 여기서, 각 구성요소의 결함율은 Table 2의 값을 사용하였다. Fig. 13의 결과로 보면, 본 논문에서 제시하는 시스템의 신뢰도가 운용시간에 따른 신뢰도 값이 이전 시스템의 신뢰도보다 우수함을 알 수 있다.

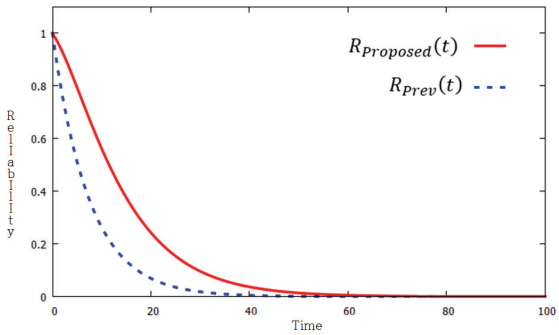

Fig. 13. numerical results of reliability analysis

4.2 실험

제시된 결함감내 PCI-Express 버스를 실험하기 위해서 device driver 소프트웨어는 50msec (20Hz) 주기[16]로 건전성 검사, 인터페이스 제어, 입출력 제어 등의 작업을 행한다. 이 중, 건전성 검사(Fig. 14)는 매 주기마다 입출력 모듈의 counter register 값을 읽고 증가시키기 전에, 이전 주기에 읽은 값과 비교하여 증가되었는지 확인한다. counter register 값이 증가되었으면 새로이 증가시킨 값을 counter register에 저장한다. 그러나 숫자가 변하지 않았으면 오류를 확인하는 작업을 수행한다. 만일 버스에 오류가 발생하면 target address 오류가 발생된 것이므로, PCI-Express system error에 대한 인터럽트가 발생한다.

본 실험에서는 이러한 시스템의 특성을 이용하여 시스템이 운영되는 중에 결함을 주입할 수 있게 하는 회로를 개발하여 사용하였다. 이 회로의 목적은 버스에 오류가 발생할 수 있게 하도록 FPGA에 실시간으로 결함을 주입할 수 있게 하는 것이다. 이 회로를 외부에서 사용하기 위해서, 입출력 모듈의 FPGA에 toggle switch를 연결하고, FPGA의 내부 회로에 reset 신호를 인가하는 기능을 구현하였다. FPGA의 Port-A와 Port-B에 서로 독립적으로 고장을 주입할 수 있으며, 스위치를 분리한 상태에서 시스템의 전원을 분리하여도 결함이 발생된 상태는 그대로 유지된다. 본 논문의 실험에서 적용된 결함주입 시점은 시스템에 전원을 인가하여 PCI-Express 채널의 결함을 등록하고 초기 상태를 설정하는 시점이다. 이 시점에서 각 FPGA 포트를 위한 결함주입 toggle switch를 인가하여 실시간으로 고장 상황을 감시하였다.

지금까지 소개한 실험 환경과 절차에서, 본 논문에서 제시된 결함감내 PCI-Express 버스의 기능 및 성능을 실험하였다. 먼저 기능 실험의 예를 들면, 시스템의 운용 중에 입출력 모듈 #1의 FPGA 포트에 결함주입을 위한 toggle switch를 인가하였을 때, target address 오류에 의한 채널 변경이 발생하였음을 확인하였다. 그리고 입출력 모듈 #1의 Port-A와 Port-B를 통해 register 값을 읽어들여서, Port-A는 고장이 발생했고, Port-B는 발생하지 않았음을 확인하였다.

다음으로 결함감내 PCI-Express 버스의 성능을 실험하기 위해서 각 작업이 수행하는데 소요된 시간을 측정하였다. 그 결과로서 버스의 고장을 감지하고 고장이 발생하지 않은 다른 포트로 register 값을 읽는데 소요된 최대시간은 [PCIe Read time+PCIe Outbound Completion Timeout 시간 + 채널전환 시간] = (4us + 1ms+1us)=1.005ms가 소요되었음을 확인하였다.

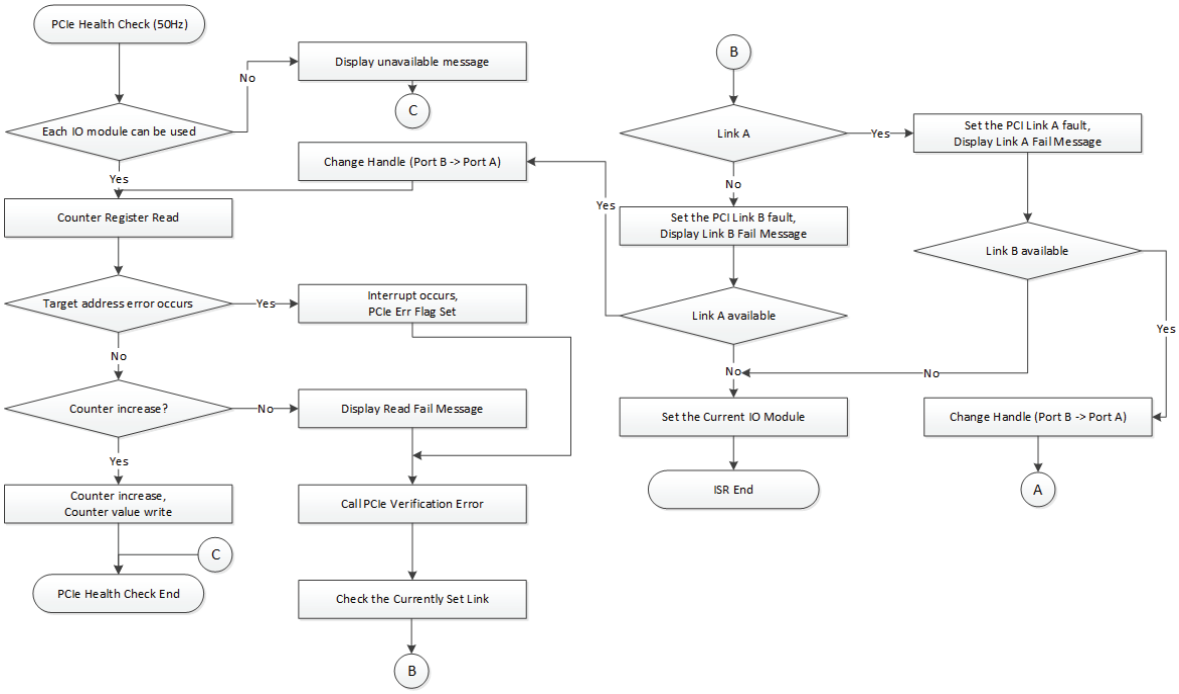

Fig. 14. health checking of fault-tolerant PCI-Express bus

V. 결 론

통합형 항공전자 시스템에서는 각 부분체계에서 공통적으로 필요한 단위 기능을 모듈화하는 임베디드 컴퓨터 시스템이 중요하기 때문에, CPU 모듈과 입출력 장치 사이에 고속으로 데이터를 전달하기 위해서 고속의 직렬 버스인 PCI-Express 버스를 사용하는 사례가 많아지고 있다.

하지만 PCI-Express 버스도 버스 자체에서 고장이 발생하면 고장의 부위에 따라 전체 모듈 또는 일부의 모듈이 사용될 수 없는 상태에 빠지게 된다. 이러한 문제는 PCI-Express 버스가 내재하고 있는 결함으로 인해서 고장이 발생해도 즉시 그 고장을 탐지하고 회피하는 결함감내(fault-tolerant) 구조로써 해결될 수 있다. 본 논문은 PCI-Express 버스의 결함감내 구조를 제공하기 위한 핵심적인 모듈로서 CPU에서 이중화된 버스와 입출력 장치의 local bus를 연결하는 인터페이스를 제시하였다.

본 논문에서 제시하는 결함감내 PCI-Express 버스는 독립된 PCI-Express switch 회로를 중복적으로 내장하여 각각의 입출력 장치에 버스의 결함을 감내하는 PCI-Express 채널을 제공한다. 입출력 장치를 위한 모듈은 switch 회로가 제공하는 중복된 채널을 FPGA 인터페이스를 이용하여 통합한다. FPGA는 고신뢰성을 제공할 뿐만 아니라, 입출력 모듈을 소형화하고 개발 비용이 절감될 수 있게 한다.

본 논문에서 제시된 결함감내 PCI-Express 버스를 신뢰도 측면에서 분석하고, 기능 및 성능의 측면에서 실험하였다. 신뢰도 분석은 결함율 모델에 따른 수치적 분석법을 사용하여 이전 시스템의 신뢰성 함수와 제시된 시스템의 신뢰도 함수를 도출하여 우수성을 분석하였다. 기능의 측면에서는, 시스템의 운용 중에 버스 구조에 결함을 주입하였을 때, 정확히 채널 변경이 발생하였음을 확인하였다. 그리고 성능의 측면에서는, 4.2절에서 실험한 결과와 같이 모든 관련 동작을 1.005 msec로 처리함으로써 항공전자 시스템에 적용하기에 적합함을 확인하였다.

후 기

이 논문은 2020년도 정부(교육부) 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. NRF-2018R1D1A3B07041838)

References

1) US DoD, *Design and Acquisition of Software for Defense Systems*, USA, 2018, p. 4.

2) PCI-SIG, *PCI Express® Base Specification*, Rev. 3.0, PCI Special Interest Group (PCI-SIG), 2010.

3) Microsemi, *PB0115 Product Brief SmartFusion2 SoC FPGA*, Microsemi, 2018, p. 8.

4) Microsemi, *RT0001 Reliability Report Microsemi FPGA and SoC Products*, Microsemi, 2018, p. 23.

5) US DoD, *MIL-STD-1553B*, USA, 1978.

6) VITA, *American National Standard for VME64*, American National Standards Institute (ANSI), 1994

7) PCI-SIG, *PCI Local Bus Specification Revision 3.0*, PCI Special Interest Group (PCI-SIG), 2002.

8) Varnavas, K., "Serial Back-Plane Technologies in Advanced Avionics Architectures," *IEEE/AIAA 24th Digital Avionics Systems Conference* (DASC), USA, 2005, pp. 12.A.1-1-3.

9) Bradford, R., et. al, "Exploring the Design Space of IMA System Architectures," *IEEE/AIAA 29th Digital Avionics Systems Conference* (DASC), USA, 2010, pp. 5.E.5-7-14.

10) D. M¨unch, et. al, "Hardware-Based I/O Virtualization for Mixed Criticality Real-Time Systems Using PCIe SR-IOV," *International Conference on Computer Science and Engineering* (ICCSE), Sri Lanka, 2013, pp. 706~713.

11) Lee, K. B., "A Study on the Transmission Technology for Reliable Mass Data Transfer between High Speed Processors for Aircraft," *Information and Control Symposium*, Korean Institute of Electrical Engineers, 2018, pp. 319~320.

12) Jeon, E. S., et. al, "Implementation of Input/Output Board with FPGAs for UAV Mission Computer," *Fall Conference of the Korean Society for Aeronautical and Space Sciences*, 2010, pp. 1501~1504.

13) Choi, E. K., Ban, C. B. and Yang, S. Y., "Development of Mission Computer Hardware for UAV," *Spring Conference of Korean Society for Aeronautical and Space Sciences*, 2013, pp. 375~378.

14) VanderLeest, S. H. and White, D., "MPSoC Hypervisor: The Safe and Secure Future of Avionics," *IEEE/AIAA 34th Digital Avionics Systems Conf.* (DASC), Prague, 2015, pp. 6B5-1-6B5-14.

15) Catton, L. W., *Avionics Display System*, US Patent No. 9,892,551-B2, GE Aviation Systems Limited, February 13, 2018.

16) Hyun, J., et al, "A Fault-Tolerant Temporal Partitioning Scheme for Safety-Critical Mission Computers," *IEEE/AIAA 31st Digital Avionics Systems Conference* (DASC), USA, 2012, pp. 1~24.

17) ARM, *AMBA(TM) Specification:* 2.0, ARM, 1999.