한국항행학회 논문지 제14권 제1호 2010년 2월

로봇의 이기종 다중 프로세서 구현을 위한 Serial RapidIO(sRIO) 분석 및 시뮬레이션

An Analysis and Simulation of sRIO for Implementation of Robot's Hetero-Multi Processor

문용선*, 노상현*, 조광훈**, 박종규***, 배영철****

Yong-Seomn Moon*, Sang-Hyun Roh*, Kwang-Hun Jo**, Jong-Kyu Park*** and Young-Chul Bae****

요 약

본 연구에서는 새로운 형태의 로봇 제어기의 구조인 이기종 멀티프로세서 제어기의 개념적인 구조를 제시하며, 제어기 내에 분산된 멀티프로세서들을 sRIO 통신을 이용하여 통합하는 구조적인 방법을 소개한다. 또한 sRIO 통신으로 통합된 이기종 멀티프로세서의 구현을 위한 방법으로 FPGA 내에 설계된 sRIO IP Core를 활용한 시뮬레이션을 수행하고 그 결과를 확인하였다.

Abstract

In this paper, we propose the structure of heterogeneous multiprocessor's concept, which is the structure of the new type of the robot controller, and we introduce an integrating structure method, which is distributed multiprocessor within controller using sRIO. We also perform the computer simulation with using the sRIO IP core which was designed within FPGA as the method for implementation of integrated heterogeneous multiprocessor by sRIO communication. Thus, we verify the result.

Key words : sRIO IP, Heterogeneous Multiprocessor, Next Robot Controller, Integration System

I. 서 론

지능형 로봇의 용도 및 기능이 다양해짐에 따라 로봇을 운용 및 제어하는 제어기의 형태 또한 점차 지능화 되어 가고 있다. 종래의 경우 로봇 개발자에 역량에 의해 독자적으로 개발한 제어기를 많이 사용하였으나 제어 성능의 한계와 신뢰성 및 범용성이 크게 떨어지는 문제점 등이 제기되었으며, 근래에 주로 사용되는 PC 기반 로봇 제어기의 구조는 처리 속도 및 기능 구현에 있어서의 장점은 있으나 실시간 성의 부재, PC 외부 자원에 따른 부하 발생, PC 보드의 크기에 따른 탑재의 어려움 등의 문제가 발생하였다. 본 논문을 통하여 제시되는 이기종 멀티프로세서는 단일 디바이스에 대한 멀티프로세서를 포함한다. 이

* 순천대학교 공과대학 전자공학과
** (주)메크로시스템 엔지니어링
*** 한국과학기술정보연구원(KISTI)
**** 전남대학교 공학대학 전기전자통신컴퓨터공학부
· 제1저자 (First Author) : 문용선
· 투고일자 : 2010년 1월 25일
· 심사(수정)일자 : 2010년 1월 27일 (수정일자 : 2010년 2월 11일)
· 게재일자 : 2010년 2월 28일

와 같은 프로세서 요소들은 신호 프로세싱, 데이터 프로세싱 및 제어 프로세싱 기능을 포함한 전체 로봇 어플리케이션에서 각각 다른 기능을 수행한다. 하나의 보드에 있는 여러개의 디바이스에서 수많은 개별적인 프로세싱 요소들을 갖춘 시스템도 존재한다. 이러한 디바이스는 데이터와 상태를 공유해 시스템 프로세싱 작업을 완성할 수 있도록 서로 간에 통신을 수행해야 한다. 많은 디바이스들은 많은 통신 라인을 갖추게 되고, 이것은 프로세서 사이를 접속하는 통신을 위한 요구조건을 작동시킨다. RapidIO(RIO)는 접속하는 문제에 대한 최상의 솔루션이라 할 수 있다. RapidIO는 임베디드 시스템에서 일반화되고 있는 업계 표준형 고속 스위치 패킷 접속이며, 접속 및 통신 모델을 요구하는 애플리케이션의 요건을 지원한다. RaipdIO의 주요 설계 목표는 시스템 상의 CPU에 가벼운 프로토콜을 제공하고, 소프트웨어의 충격을 제한하여, 내부 통신에 집중한다. 폭넓게 호환되는 아키텍처는 칩 간 통신 뿐만 아니라 보드 간 통신을 위해 증가된 대역폭에서 신뢰성 높은 고성능, 패킷 스위치형 기술을 제공한다. [1][2]

이에 본 연구에서는 새로운 로봇 제어기의 형태라 할 수 있는 이기종 멀티프로세서 제어기 내의 분산된 프로세서들 간의 통합 인터커네트용 통신 솔루션으로서 Serial RapidIO(sRIO)를 제시하며 이를 위한 세부적인 시뮬레이션을 수행하였다. 본 논문의 구성은 1장의 서론을 시작으로 2장에는 sRIO를 이용한 이기종 멀티 프로세서의 통합 구조를 제시하였으며, 3장에서는 sRIO 구현을 위한 시뮬레이션을 수행하였으며, 4장의 결론 순으로 기술을 하였다.

Ⅱ. sRIO를 이용한 이종 다중프로세서 통합

이기종 멀티프로세서 제어기는 DSC, ARM, FPGA 등의 서로 다른 프로세서를 분산 배치하고 이들 사이의 고속의 데이터 접속용 통신을 이용한 통합을 수행함으로서 고속의 분산 처리를 실현하였으며, 메인 처리 프로세서의 기종을 다양화 으로서 로봇의 어플리케이션의 형태에 따른 최적화된 제어기의 구현이 가능할 뿐만 아니라 종래의 로봇 개발자의 접근을 용이성을 제공한다. 그림 1은 이기종 멀티 프로세서 형 로봇 제어기의 개념적인 구조를 나타낸다.

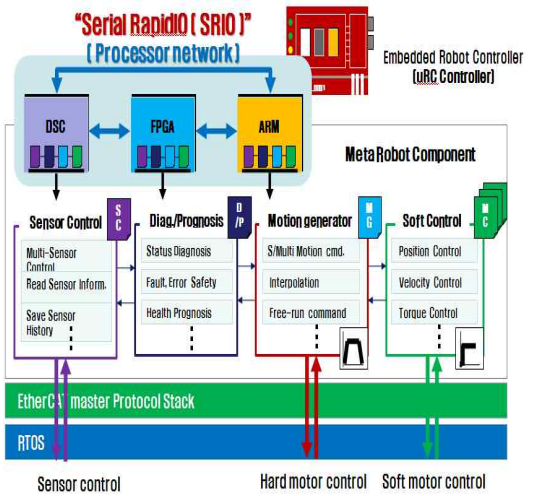

그림 1. 이기종 멀티프로세서 기반의 임베디드 로봇 제어기 구조
Fig. 1 The Structure of embedded robot controller based heterogeneous multiprocessor

이기종 멀티 로세서의 구현을 위한 핵심은 각각의 프로세서들이 얼마나 효율적으로 분산된 자원을 서로 자유롭게 공유할 수 있으며 또한 얼마나 고속으로 처리된 데이터를 교환할 수 있는지에 달려 있다고 할 수 있다. 이러한 요구조건을 만족하기 위해 필요한 것이 고속으로 프로세서들간의 접속이 가능한 통신 방식의 설계 및 구현이다.

Serial RapidIO(sRIO)는 고성능 연산 처리 장치나 백 플레인 같은 고성능 임베디드 기기응용을 주요 목표로 하는 표준화된 입출력 통신 방식으로서 Gbps 레벨의 고속의 전송속도, 다양한 칩-투-칩 연결 토폴로지 지원, 신호의 안전성 등 이기종 멀티 프로세서들 간의 상호연결 문제에 대한 최상의 통신 솔루션이라 할 수 있다. 다음의 그림 2~4는 sRIO의 링크 구성의 변경을 통한 다양한 형태의 이기종 멀티 프로세서의 통합 토폴로지 구조를 나타낸다. [1][2]

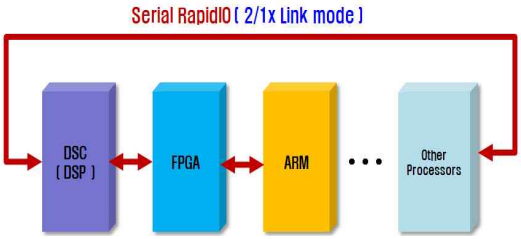

그림 2. sRIO 2/1x 링크 모드를 이용한 이기종 멀티프로세서 간의 링 토폴로지 구조
Fig. 2 The structure of ring topology between heterogeneous multiprocessor using sRIO 2/1x link mode

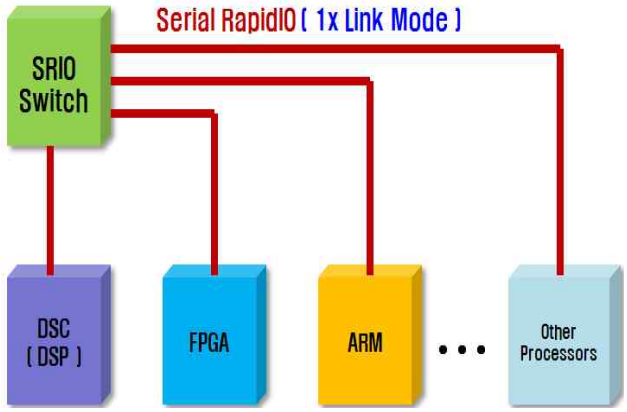

그림 3. sRIO 1x 링크 모드와 sRIO 스위치를 이용한 이기종 멀티프로세서 간의 트리 토폴로지 구조
Fig. 3 The structure of tree topology between heterogeneous multiprocessor using sRIO 1x link mode and sRIO switch

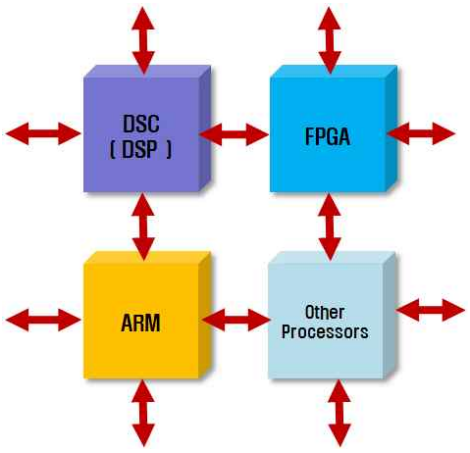

그림 4. sRIO 4/1x 링크 모드를 이용한 이기종 멀티프로세서 간의 메쉬 토폴로지 구조
Fig. 4 The structure of mesh topology between heterogeneous multiprocessor using sRIO 4/1x link mode

Ⅲ. sRIO 시뮬레이션

본 장에서는 이기종 멀티프로세서의 통합을 위한 접속 통신 솔루션으로서 사용될 sRIO 통신에 대한 시뮬레이션을 수행하였다. sRIO 통신 시뮬레이션을 위한 환경은 Altera Stratix II GX FPGA 내에 설계된 sRIO IP Core를 기반으로 수행하였으며 시뮬레이션의 실행 및 결과는 FPGA용 시뮬레이션 툴인 ModelSim-Altera를 사용하였다. Altera 사는 FPGA 내의 sRIO IP Core 설계 및 시뮬레이션을 위한 MegaCore function을 제공하고 있으며 이는 그림 5과 같이 구성된 sRIO 시뮬레이션 구조를 제공하고 있다. [4]

sRIO 시뮬레이션의 세부적인 수행은 표준 sRIO 스펙에서 정의하는 오퍼레이션들 중 가장 많이 사용이 되는 IO 오퍼레이션들을 중심으로 이루어졌으며 그 중에서도 표1에서와 같이 정의한 4가지 오퍼레이션들을 중심으로 수행을 하였다.

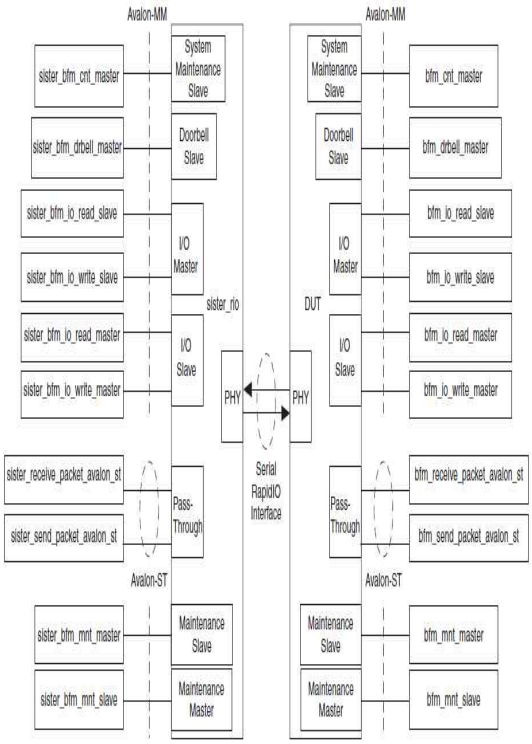

그림 5. sRIO 시뮬레이션 구조
Fig. 5 The structure of sRIO simulation

표 1. 시뮬레이션을 수행할 sRIO 표준 오퍼레이션 및 세부 트랜잭션
Table 1. sRIO standardization and detail transaction will perform for computer simulation

Operation	Transaction
Read Operation	NREAD, RESPONSE
Write Operation	NWRITE
Write with response Operation	NWRITE_R, RESPONSE
Maintenance Operation	MAINTENANCE

3-1 READ 시뮬레이션

Read 오퍼레이션은 특정된 어드레스로부터 데이터를 읽기 위해 사용되는 명령으로서 sRIO 네트워크 상에서 분산된 장치들로부터 특정된 정보의 수신이 필요한 경우에 사용이 된다. Read 오퍼레이션은 NREAD 트랜잭션과 RESPONSE 트랜잭션으로 구성이 된다.[4][5]

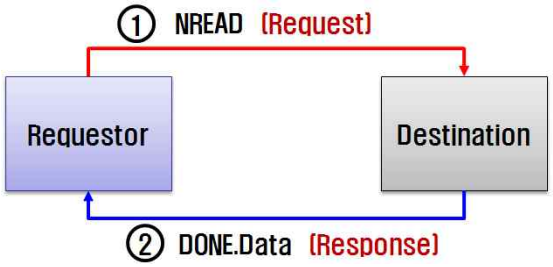

그림 6. Read 오퍼레이션 개념도
Fig. 6 Block diagram of read operation

NREAD 시뮬레이션은 Avalon MM I/O Interface를 통하여 NREAD Request를 수행하며, Response 된 결과를 다시 수행하는 방식으로 수행이 되었다. NREAD 시뮬레이션에서는 FPGA로 구현된 로봇 제어 프로세서에서 sRIO 통합 네트워크를 통하여 분산된 장치(메모리, DSC, Arm 등)로부터 데이터를 읽어들이는 과정을 시뮬레이션 하였다. NREAD 트랜잭션에 대한 시뮬레이션 구조를 그림 7에 나타내었으며 ModelSim 웨이브 파형의 출력 결과를 그림 8에 나타내었다.

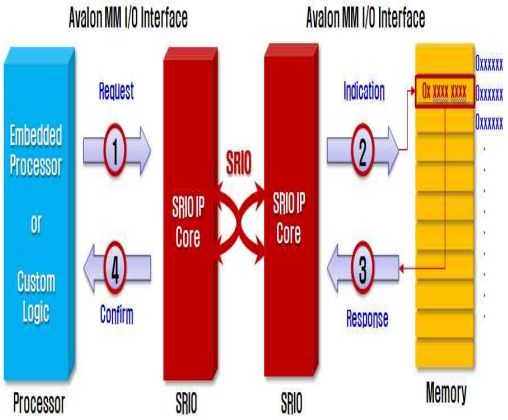

그림 7. NREAD 시뮬레이션 구조
Fig. 7 The structure of NREAD simulation

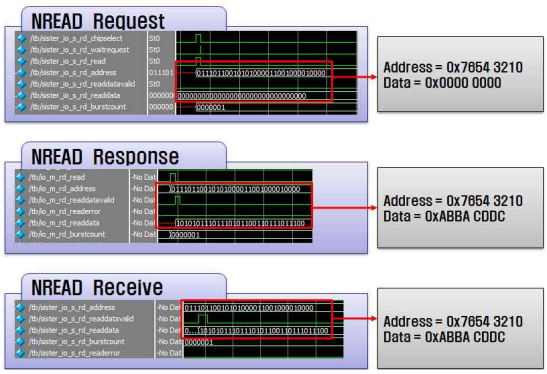

그림 8. NREA 시뮬레이션 웨이브 출력 결과
Fig. 8 The output result of NREA simulation wave

각각의 시뮬레이션 구간별 시뮬레이션 소요 시간에 대한 결과는 그림 9와 같다.

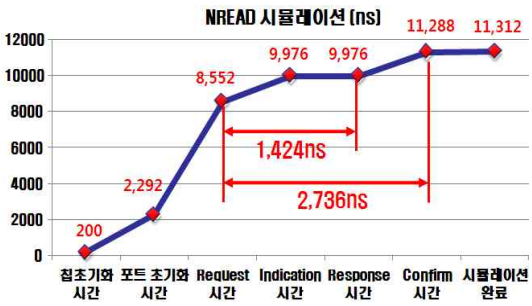

그림 9. NREAD 시뮬레이션 시간 결과
Fig. 9 The time result of NREAD simulation

3-2 Write 시뮬레이션

Write 오퍼레이션은 특정된 어드레스로부터 데이터를 쓰기위해 사용되는 명령으로서 sRIO 네트워크 상에서 분산된 장치들에 특정된 정보의 전송이 필요한 경우에 사용이 된다. Write 오퍼레이션의 경우 Response 패킷이 존재하지 않으므로 데이터의 신뢰성이 필요한 어플리케이션 보다는 고속의 데이터 전송이 필요한 어플리케이션에 더 적합하다고 할 수 있다. 이러한 Write 오퍼레이션에는 NWRITE 트랜잭션이 있다.[4][5]

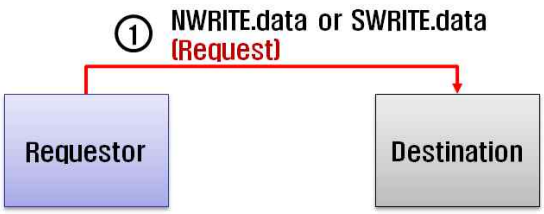

그림 10. Write 오퍼레이션 동작 개념도
Fig. 10 Block diagram of write operation

본 연구에서는 Write 오퍼레이션의 시뮬레이션의 경우 보다 유연한 전송 구조를 지원하는 NWRITE 트랜잭션에 대한 시뮬레이션을 수행하였다. NWRITE 시뮬레이션에서는 FPGA로 구현된 로봇 제어 프로세서에서 sRIO 통합 네트워크를 통하여 분산된 장치(메모리, DSC, Arm 등)에 데이터를 쓰는 과정을 시뮬레이션 하였다. NWRITE 시뮬레이션은 Avalon MM I/O Interface를 통하여 NWRITE Request 및 Request 된 정보를 수신하는 과정 순으로 수행하였다. NWRITE 시뮬레이션 구조를 그림 11에 나타내었다.

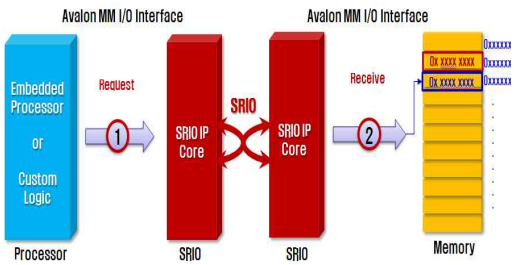

그림 11. NWRITE 시뮬레이션 구조
Fig. 11 The structure of NEWITE simulation

NWRITE 트랜잭션에 대한 ModelSim 웨이브 파형 출력 결과는 그림 12~13과 같다. 그림 12의 경우 송신 측에서 전송하는 NWRITE 데이터 정보를 나타내며, 그림 13의 경우 수신 부에서 수신한 정보의 결과를 나타낸다.

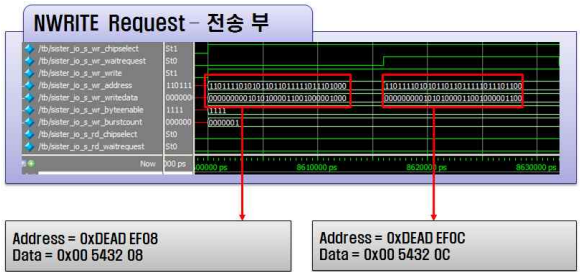

그림 12. NWRITE 시뮬레이션 웨이브 송신부 출력 결과
Fig. 12 The output result of NWRITE simulation transmitter wave

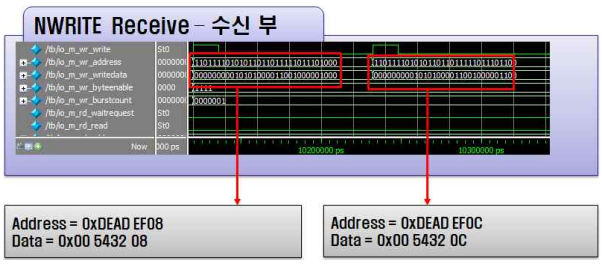

그림 13. NWRITE 시뮬레이션 웨이브 출력 결과
Fig. 13 The output result of NWRITE simulation receiver wave

NWRITE 시뮬레이션 수행 중 각 구간별로 측정된 시뮬레이션 소요 시간에 대한 결과는 그림 14와 같다.

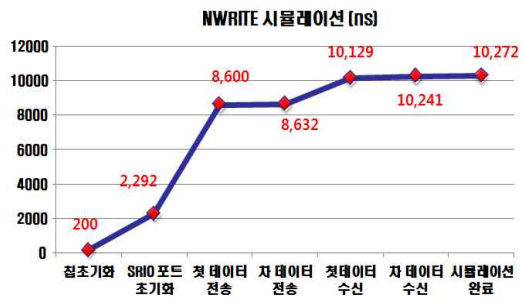

그림 14. NWRITE 시뮬레이션 시간 결과
Fig. 14 The time result of NWRITE simulation

3-3 WRITE with Response 시뮬레이션

Write with Response 오퍼레이션은 Write 오퍼레이션과 같이 특정된 어드레스로부터 데이터를 쓰기위해 사용되는 명령이다. 그러나 Write with Response 오퍼레이션의 경우에는 Request 된 패킷에 따른 Response 패킷이 존재하므로 Write 오퍼레이션 보다는 데이터의 신뢰성이 요구되는 어플리케이션에 최적으로 사용될 수 있다. 이러한 Write with Response 오퍼레이션을 수행하는 트랜잭션에는 NWRITE_R 트랜잭션 있다. [4][5]

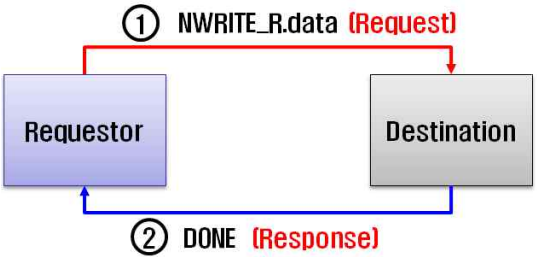

그림 15. Write with Response 오퍼레이션
Fig. 15 Write with Response operation

NWRITE_R 시뮬레이션은 Avalon MM I/O Interface를 통하여 NWRITE_R Request를 수행하며, 수신 부 측에서 데이터 수신 후 전송하는 Response 패킷을 다시 수신하는 방식으로 수행이 되었다. NWRITE 시뮬레이션의 경우에는 NWRITE 트랜잭션에 비해 다양한 타입의 데이터 방식을 지원하지는 않지만 핸드세이크 방식의 데이터 교환 구조로 인하여 중요 데이터의 전송시 유용하게 적용할 수 있다.

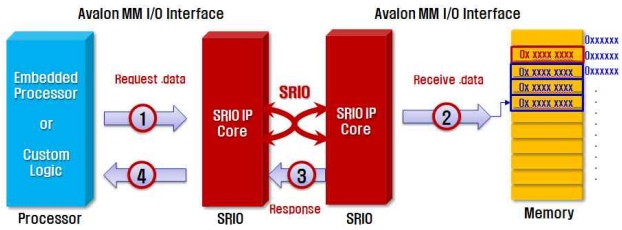

그림 16. NWRITE_R 시뮬레이션 구조
Fig. 16 The structure of NWRITER_R simulation

NWRITE_R 트랜잭션에 대한 ModelSim 웨이브 파형 출력 결과는 그림 17~18과 같다. 그림 17의 경우 송신 측에서 전송하는 NWRITE_R 데이터 정보를 나타내며, 그림 18의 경우 수신 부에서 수신한 정보의 결과를 나타낸다.

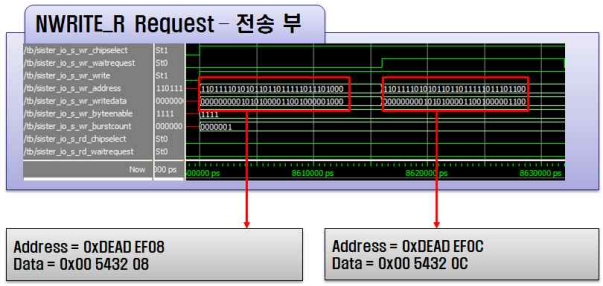

그림 17. NWRITE_R 송신부 결과
Fig. 17 Result of NWRITE_R transmitter

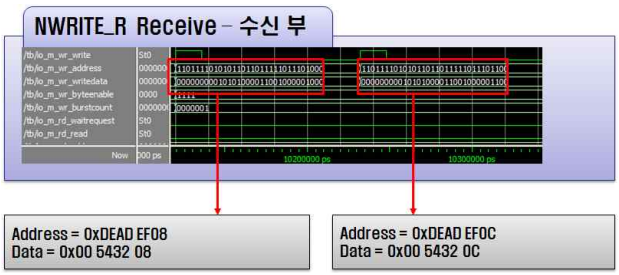

그림 18. NWRITE_R 수신부 결과
Fig. 18 Result of NWRITE_R receiver

그림 19는 NWRITE 트랜잭션과 NWRITE_R 트랜잭션과의 차이를 나타내는 Response 신호의 결과 유무를 비교한 결과이다.

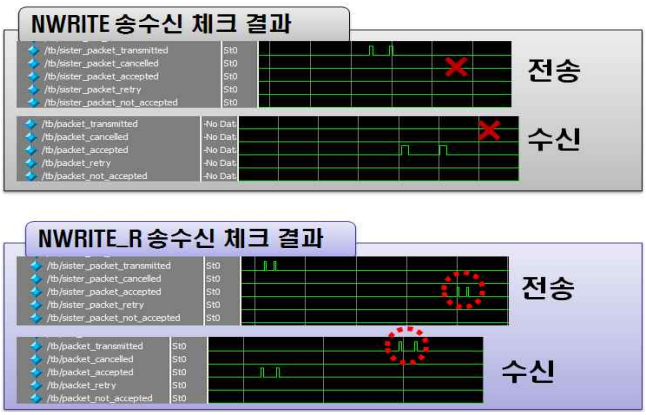

그림 19. NWRITE과 NWRITE_R 간의 Response 신호 비교
Fig 19 Comparision of response singnal between NWRITE and NWRITE_R

NWRITE_R 시뮬레이션 수행 중 각 구간별로 측정된 시뮬레이션 소요 시간에 대한 결과는 그림 20과 같다

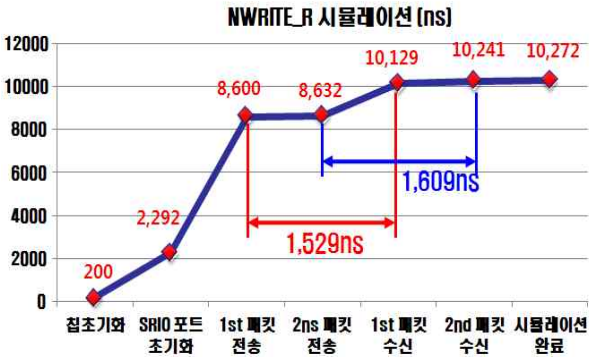

그림 20. NWRITE_R 시뮬레이션 시간 결과

Fig. 20 The time result of NWRITE_R simulation

3-4 MAINTENANCE 시뮬레이션

Maintenance 오퍼레이션은 특정 레지스터의 정보를 읽어 들이거나 특정 레지스터에 값을 쓰고자 할 때 사용되는 명령으로서 sRIO 네트워크 상에 분산된 장치들의 동작 변경을 위한 레지스터 세팅이나, 진단 및 에러 정보를 전송하는 데 주로 사용되는 방식이다. 이러한 Maintenance 오퍼레이션에는 MAINTENANCE Write, MAINTENANCE Read, MAINTENANCE Port Write 등의 트랜잭션이 있다. [4][5]

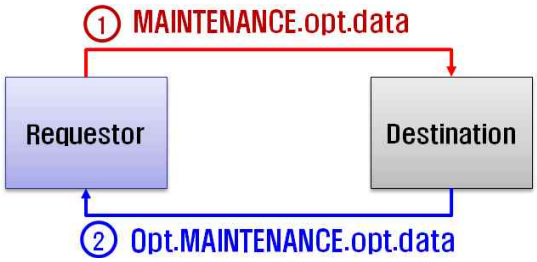

그림 21. MAINTENANCE 오퍼레이션

Fig. 21 MAINTENANCE operation

본 연구에서는 MAINTENANCE 오퍼레이션 중 분산된 장치의 레지스터에 값을 쓰기위해 사용되는 MAINTENANCE Write 트랜잭션에 대한 시뮬레이션을 수행하였다. MAINTENANCE Write 트랜잭션은 Avalon MM Maintenance Interface를 통하여 MAINTENANCE Write Request를 수행하며, 수신 부 측에서 데이터 수신 후 전송하는 Response 패킷을 다시 수신하는 방식으로 수행이 하였다.

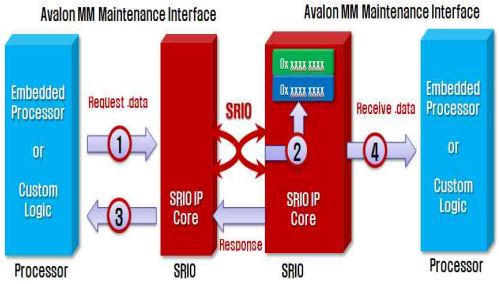

그림 22. MAINTENANCE Write 시뮬레이션 구조

Fig. 22 The structure of MAINTENANCE Write simulation

MAINTENANCE Write 트랜잭션에 대한 ModelSim 웨이브 파형 출력 결과는 그림 23과 같다. 그림 23의 상단의 웨이브 출력 결과의 경우 송신 측에서 전송하는 MAINTENANCE Write 데이터 정보를 나타내며, 하단의 웨이브 출력 경과는 수신 부에서 수신한 정보의 결과를 나타낸다.

그림 23. MAINTENANCE Write 송신부 및 수신부 결과

Fig. 23 The result of MAINTENANCE Write's transmitter and receiver simulation

MAINTENANCE Write 시뮬레이션 수행 중 각 구간별로 측정된 시뮬레이션 소요 시간에 대한 결과는 그림 24~25과 같다

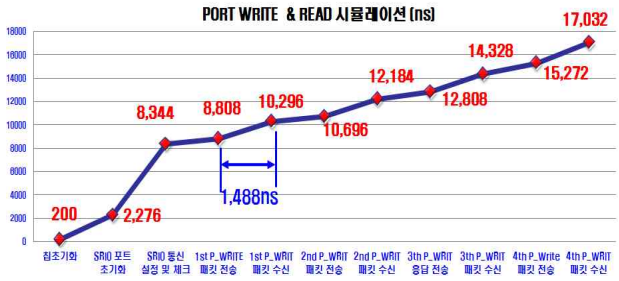

그림 24. MAINTENANCE Write 시뮬레이션 시간 결과 1
Fig. 24. The time result 1 of MAINTENANCE Write simulation

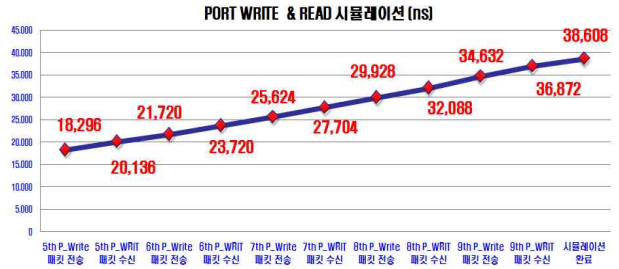

그림 25. MAINTENANCE Write 시뮬레이션 시간 결과 2
Fig. 25. The time result 2 of MAINTENANCE Write simulation

Ⅳ. 결 론 및 향후 과제

본 연구에서는 기존의 로봇 제어기의 문제점으로 제기되었던 공용성의 부재, 단일 프로세서의 적용에 따른 분산처리 부재 및 프로세서 과부하 등의 문제점을 해결하기 위하여 새로운 형태의 로봇 제어기의 구조인 이기종 멀티 프로세서 제어기의 구조를 제시하였다. 이러한 이기종 멀티 프로세서 제어기는 다양한 로봇 유저의 제어기 사용을 가능하게 하면서도 로봇의 기능 및 행위에 따른 태스크를 분산처리 함으로서 로봇 시스템의 전체적인 성능 향상시킬 수 있다. 이를 위하여 본 연구에서는 DSC, ARM, FPGA 등을 이용하여 로봇 제어기를 구성하였으며 제어기 내에 분산된 멀티 프로세서들을 sRIO 통신을 이용하여 통합하는 구조적인 방법을 소개하였다. 그리고 sRIO 통신으로 통합된 이기종 멀티 프로세서의 구현을 위한 일환으로 FPGA 내에 설계된 sRIO IP Core를 활용한 시뮬레이션을 수행하였다. 세부적인 시뮬레이션의 수행은 sRIO 표준 상에서 기술하고 있는 오퍼레이션들 중 이기종 멀티 프로세서 시스템 구현 시 주로 사용될 NWRITE, NREAD, NWRITE_R, MAINTENANCE의 4가지 오퍼레이션들에 대한 시뮬레이션을 수행하였으며 정상적인 시뮬레이션 결과를 얻을 수 있었다. 향후에는 실제 하드웨어로 구현된 이기종 멀티 프로세서 보드 상에서 sRIO 통신에 대한 동작 실험 및 종합적인 성능을 검증하고자 한다.

감사의 글

본 연구는 교육과학기술부와 한국산업기술진흥원의 지역혁신인력양성사업으로 수행된 연구결과임.

참 고 문 헌

[1] Texas Instrument, 임베디드 시스템의 sRIO 아키텍처, 2008
[2] RapidIO Trade Association, RapidIO Technology Overview and Applications, 2005
[3] RapidIO Trade Association, System Interconnect Fabrics : Ethernet versus RapidIO Technology, 2007
[4] Altera, RapidIO MegaCore Function User Guide, 2009
[5] RapidIO Trade Association, RapidIO Standard Specification Part1~12, 2009
[6] Tundra, Tsi574 Serial RapidIO Switch User Manual, 2007
[7] IDT, Serial RapidIO Capabilities Enable DSP Cluster Applications, 2006

문 용 선(文庸善)

1983년 2월 : 조선대학교 전자공학과(공학사)
1989년 2월 : 조선대학교 대학원 전자공학과(공학박사)
1992년 ~ 현재 : 순천대학교 정보통신공학부 교수 및 레드원테크놀러지(주) 기술이사

관심분야 : 산업통신망 및 로봇, 실시간 모션 제어

노 상 현(盧相賢)

2007년 2월 : 순천대학교 전자공학과(공학사)
2009년 2월 : 순천대학교 전자공학과(공학석사)
2009년 ~ 현재 : 순천대학교 전자공학과(공학박사 재학 중)

관심분야 : 로봇 제어, 모터 제어, 산업통신망

조 광 훈(趙廣勳)

2003년 2월 : 순천대학교 전자공학과 졸업(공학사)
2003년 2월 (주)메크로 시스템 엔지니어링 입사

관심분야 : 자동화 시스템, 산업통신망

박 종 규(朴 宗圭)

1984년 2월 : 중앙대학교 전자공학과(공학사)
19908년 2월 : 중앙대학교 전자공학과(공학석사)
1991년 ~ 현재 : 한국과학기술 정보연구원 선임연구원

배 영 철(裵英哲)

1984년 2월 : 광운대학교 전기공학과(공학사)
1986년 2월 : 광운대학교대학원 전기공학과(공학석사)
1997년 2월 : 광운대학교대학원 전기공학과(공학박사)
1986년 ~ 1991년 : 한국전력공사
1991년 ~ 1997년 : 산업기술정보원 책임연구원
1997년 ~ 2006년 : 여수대학교 전자통신전기공학부 부교수
2006년 ~ 현재 : 전남대학교 전기전자통신컴퓨터공학부 교수
관심분야 : Chaos Control and Chaos Robot, Robot control etc.