ISSN 1975-8359(Print) / ISSN 2287-4364(Online)
The Transactions of the Korean Institute of Electrical Engineers Vol. 66, No. 1, pp. 186~193, 2017
http://doi.org/10.5370/KIEE.2017.66.1.186

실시간 이더넷 기반의 한국형 오픈소스 모션 시스템 개발 및 분석

Development and Analysis of Korea Open Source Motion System based on Real-Time Ethernet

임 선† · 이 승 용* · 김 지 현* · 정 일 균*
(Sun Lim · Seung-Yong Lee · Ji-Hyun Kim · Il-Kyun Jung)

Abstract - KOSMOS is Korea Open Source MOtion System which is developed based on general purpose hardware and open source software. It is aiming at IEC 61131-3 standard. Real-time ethernet has several advantages for motion control system and distributed control system. So, considering this advantages, KOSMOS has the network interface made up of Real-time ethernet, EtherCAT. In this paper, we explain the KOSMOS platform, the performance for real-time task and show the real case applying KOSMOS platform in manipulator control system.

Key Words : Motion control system, Real-time ethernet, EtherCAT, IEC 61131-3

1. 서 론

현재 우리나라의 제조업은 산업 자동화와 관련된 기술 개발 추세에 비해 부분적으로 적용되어 실직적인 공장 자동화를 이루지 못하고 있는 실정이다. 이러한 문제의 원인으로는 기존에 사용하고 있는 장비와의 호환성, 확장성 및 공장 자동화를 위한 막대한 초기 비용 등이 있을 것이다. 이와 같은 문제를 해결하기 위한 방안으로 x86, ARM 프로세서와 Linux 등의 범용 하드웨어 및 소프트웨어를 이용한 모션 제어 시스템에 대한 연구[6]가 활발히 이루어지고 있다. 또한 오픈 소스를 이용한 모션 제어 시스템 개발을 통해 비용성, 호환성, 확장성 등에 대한 문제점을 해결할 수 있을 것으로 보인다.

과거에는 특정 필드버스만으로 제어 시스템이 구성되어 시스템의 호환성, 낮은 프로토콜 성능과 국제 표준 준수의 미흡 등 다양한 한계점을 지니고 있다. 이와는 별개로 최근에 들어 통합 제어 시스템의 발전에 따라 분산 제어 시스템 개발에 대한 요구가 증대되고 있다. 이를 해결하기 위해 실시간 이더넷 기반의 산업용 네트워크와 관련된 기술 개발[2,3,5]이 활발히 진행되고 있다. 그 중, 독일의 Beckhoff에서 개발한 EtherCAT[1]은 다양한 토폴로지 형태의 네트워크를 지원하기 때문에 분산 네트워크 시스템의 지원이 용이하며 실시간 시스템 지원, 높은 대역폭, 고정밀 동기화 기법 등의 장점을 지니고 있다. 위와 같은 이유로 모터 구동용 서보 드라이브에서 EtherCAT을 지원하는 장치가 출시됨에 따라 공장 자동화 및 로봇 시스템에서도 점차적으로 그 적용성이 확대 적용되고 있다. 따라서 이를 지원하기 위해 오픈 소스의 EtherCAT Master Stack을 포함한 모션 시스템 개발이 필요한 실정이다[18].

본 논문에서는 IEC 61131-3 표준[16]을 지향하며 EtherCAT Master Stack을 지원하는 오픈소스 기반의 한국형 모션 제어 시스템 개발에 대해 제시한다. 사용자는 PLCOpen 통합개발환경인 KOSMOS(Korea Open Source MOtion System, 이하 KOSMOS) IDE를 이용하여 원하는 PLC 프로그램을 작성하거나 제어 시스템 개발이 가능하다. 또한 사용자의 편의를 위해 HMI(Human Machine Interface, 이하 HMI)와 OPC UA(Open Platform Communications Unified Architecture, 이하 OPC UA) 확장 소프트웨어 개발을 진행하였다. 또한 모션 제어기 개발을 위해 ARM 프로세서 및 오픈 소스의 실시간 운영체제 기반의 모션 제어기인 GMC(General Motion Controller, 이하 GMC)와 모터 드라이브 및 외부 I/O 모듈과의 연결을 위해 EtherCAT 실시간 산업용 이더넷을 채택하여 모션 시스템을 구성하였다.

본 논문의 2장에서는 모션 제어 시스템과 관련된 연구 배경에 대해 서술하며 3장에서는 한국형 모션 제어 시스템을 제시한다. 4장에서는 한국형 모션 제어 시스템과 관련된 성능 평가 및 실증 결과에 대해 제시 후 마지막으로 결론 및 향후 연구에 대해 서술한다.

2. 연구배경

최근 산업 자동화 및 지능형 공장과 관련된 많은 이슈가 발생하여 이를 개선하기 위한 연구 개발이 진행되고 있다. 이와 함께 산업 자동화의 핵심인 모션 제어 시스템과 그 시스템을 구성하는 필드버스와 관련된 연구 또한 활발히 진행되고 있다. 모션 제어 시스템은 일반적으로 모션 제어기, 모터 드라이브, I/O 모듈 등으로 구성된다. 과거에는 Profibus, CC-Link, CANOpen 등 특정 필드버스만으로 제어 시스템이 구성되었다. 그러나 이는 시스템

† Corresponding Author : Korea Electronic Technology Institute
E-mail:mickey3d@keti.re.kr
* Korea Electronic Technology Institute
Received : December 2, 2016; Accepted : December 22, 2016

호환성, 낮은 프로토콜 성능과 국제 표준 준수의 미흡 등 다양한 한계점을 지니고 있다. 최근에는 이를 해결하기 위해 실시간 이더넷 기반의 산업용 네트워크와 관련된 기술 개발이 진행되고 있다. 그 중 Ethernet/IP, EtherCAT, POWERLINK[4]가 많은 분야에서 사용하고 연구되고 있다. 특히 EtherCAT은 독립 성능 평가 연구와 POWERLINK와의 비교 성능 연구에서 높은 네트워크 성능을 가짐을 알 수 있었다.

최근 모션 제어 시스템과 관련된 연구 중에 오픈소스를 이용한 시스템 개발 및 성능 평가와 관련된 많은 연구들이 진행되어 왔다. 과거에는 특정 제조사의 모터, 모터 드라이브 등의 하드웨어와 소프트웨어를 이용하여 제조 시스템을 구성해왔다. 제조사에 의존적인 소프트웨어 코드의 경우 범용성이 현저히 떨어지며 특히 제조사에서 제공하는 특별 함수를 사용하는 경우 그 내용 구성 형태에 대해서 사용자가 알 수 없어 활용성 또한 매우 제한적이다. 이러한 요인으로 인해 최근 다양한 제조업 시스템의 이식성과 확장성 그리고 비용성 측면에서의 장점을 갖추기 위해 오픈소스를 이용한 모션 제어 시스템 연구 개발이 요구되고 있다. 특히 x86, ARM 프로세서와 Linux 등의 범용 하드웨어 및 소프트웨어를 이용한 시스템 개발과 연구가 활발히 진행되고 있다. 그 중 본 논문에서 채택한 Xenomai RTOS[7]는 범용 Linux를 기반으로 개발된 오픈소스 운영체제이다. 많은 분야에서 사용되는 상용 RTOS인 VxWorks와 Linux 기반의 RTAI, Linux-RT와의 비교 성능 평가[8-11]에서 Xenomai 운영체제가 네트워크 지연과 지터 부분에서 더 높은 실시간 성능을 보였다. 또한 최근에는 다중 프로세서를 이용한 기존 모션 제어 시스템과 로봇 시스템의 실시간 성능 개선을 위한 연구[12,13]가 진행되고 있다. 기존 하나의 프로세서에서 동작되던 제어 시스템을 시스템 특성에 따라 여러 개의 태스크로 구분하여 동작하도록 개발하였고, 이후 여러 프로세서를 이용한 수행이 가능해졌다. 그 결과, 전체 시스템을 여러 시스템으로 부분화함으로써 다른 태스크의 간섭을 줄이고 성능 최적화를 통한 전체 시스템의 오버헤드를 줄일 수 있으며 또한 사용자의 개발 용이성의 장점을 가질 수 있다.
본 논문에서는 생산 자동화 현장을 위한 한국형 오픈소스 모션 제어 시스템의 개발에 대해 제시한다. 범용 Linux 기반의 실시간 운영체제인 Xenomai를 이용하여 x86, ARM 범용 하드웨어 기반의 모션 제어기를 구성한다. 또한 통합개별 환경 지원을 위하여 Beremiz 오픈소스[14]를 이용한 KOSMOS IDE를 개발하고 지원한다. 모션 제어기의 역할 중 하나인 모터 드라이브 및 외부 I/O 제어를 위해 EtherCAT 실시간 산업용 이더넷을 채택하였다. EtherCAT 지원을 위해 Etherlab[15]에서 제공하는 IgH EtherCAT Master Stack 오픈소스를 이용하여 KOSMOS IDE 모션 제어기에서 구동되도록 개발하였다.

3. 한국형 모션 제어 시스템의 설계 및 구현

3.1 한국형 오픈소스 기반 모션 제어 시스템

본 논문에서는 최근 오픈소스 기반의 모션 제어 시스템 개발 요구에 맞춰 IEC 61131-3[16] 표준을 지향으로 하는 오픈소스 기반의 모션 제어 플랫폼 개발 및 분석에 대해 제시한다. 최근 미국, 유럽 등 선진국에서는 IEC 61131-3 기반의 모션 제어 시스템이 연구 및 개발이 진행되고 있다. 이런 흐름에 맞춰 본 논문에서는 오픈소스를 이용한 한국형 모션 제어 시스템을 제안하고 상용 시스템과 비교하여 동일한 성능뿐만 아니라 호환성 및 확장성의 장점을 있음을 증명하였다.

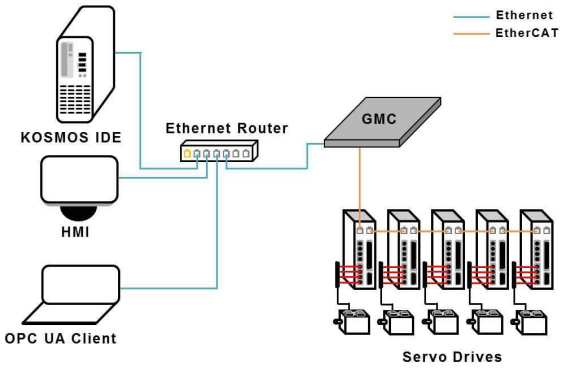


그림 1 한국형 모션 제어 시스템 전체 개략도
Fig. 1 Whole schematic diagram of KOSMOS

그림 1은 본 논문에서 제시하는 KOSMOS 전체 개략도의 한 예이다. KOSMOS IDE는 PLCOpen 통합개발환경으로 현재 오픈소스로 공개되는 Beremiz 통합개발환경 소프트웨어를 활용하였다. 이를 이용하여 사용자는 IEC 61131-3 표준에 따라 원하는 PLC 프로그램을 작성하고 제어 및 운용 프로그램을 구현할 수 있다. 산업 자동화에서 사용되는 모션 제어 시스템은 다양한 사용자 요구가 발생함에 따라 KOSMOS IDE 또한 다양한 확장 개발이 가능하다. 본 논문에서는 이러한 기능을 이용하여 OPC UA와 HMI, EtherCAT 및 모션 라이브러리를 위한 확장 소프트웨어를 개발하였다. GMC는 저가형 임베디드 플랫폼에서 고성능의 산업용 PC까지 사용자가 원하는 하드웨어를 이용하여 개발될 수 있는 범용 모션 제어기를 의미한다. 본 논문에서는 ARM 범용 프로세서를 이용하여 GMC를 개발하였으며 이를 이용하여 성능 시험을 진행하였다. 그 외로 사용자 편의를 위해 HMI와 OPC UA 기능을 제공하여 모션 제어 시스템을 모니터링 할 수 있도록 KOSMOS 플랫폼을 개발하였다.

그림 2는 KOSMOS 플랫폼 중 GMC에 탑재되는 소프트웨어 구조도를 나타낸다. GMC는 위에서 서술한 것과 같이 x86 기반의 산업용 PC뿐만 아니라 ARM 프로세서 기반의 범용 임베디드 플랫폼에서도 동작시킬 수 있다. 또한 GMC에는 외부 인터페이스와 통신하기 위해 Ethernet 및 서보 드라이브, I/O 등 EtherCAT Slave 통신을 위한 디바이스 드라이버가 각각 존재하게 된다. 본 논문에서는 산업용 PC와 비교하여 낮은 성능을 가지는 ARM 임베디드 플랫폼에서도 KOSMOS 플랫폼의 이식성과 실시간 성능을 증명하기 위해 별도로 GMC 하드웨어를 제작하였으며 ARM 프로세서 기반에서 동작할 수 있도록 관련 디바이스 드라이버 제작 등의 추가적인 작업을 진행하였다. GMC는 모션 제어기 역할을 담당하므로 실시간 성능이 매우 중요함을 알 수 있다. 따라서 범용 Linux 기반의 실시간 RTOS인 Xenomai RTOS에서 동작되

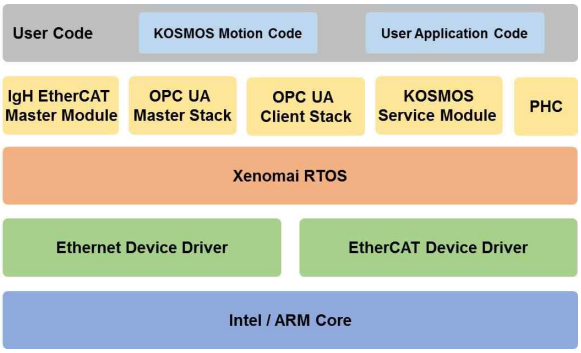


그림 2 GMC 소프트웨어 구조도
Fig. 2 Structure of GMC software

도록 개발하였다. 2절에서 설명한 것과 같이 Xenomai RTOS가 높은 실시간 성능을 가짐을 알 수 있으며 본 논문의 Xenomai 운영체제 적합성을 확인할 수 있었다. 또한 EtherCAT Master를 지원하기 위해 IgH EtherCAT Master Stack 오픈소스를 채택하였다. 추가적으로 프로세스를 관리하기 위한 스크립트를 개발하여 멀티코어 기반의 KOSMOS 플랫폼이 동작되도록 하였다. 이와 관련하여 자세한 사항은 본 절 이후에 서술한다.

3.2 GMC 소프트웨어 개발

3.1절에서 제시한대로 본 논문에서는 KOSMOS 플랫폼의 이식성, 성능 평가 등을 위해 ARM 프로세서 기반의 GMC 하드웨어를 채택하였다. 그러나 KOSMOS IDE를 개발하기 위해 사용된 Beremiz 소프트웨어는 현재 x86 프로세서 기반의 Win32 또는 Linux 운영체제만을 지원하고 있다. 따라서 ARM 프로세서 기반의 Xenomai 운영체제에서 KOSMOS 응용이 실행되기 위해 Linaro-4.6.3-ubuntu5 ARM 크로스 컴파일러가 제공되도록 KOSMOS IDE 소프트웨어 개발을 진행하였다.

GMC의 실시간 성능을 만족하기 위해 이식된 Xenomai RTOS는 범용 Linux OS에 사용하기 위해 추가 개발이 필요하다. 첫 번째로 Linux 커널과 독립적으로 실시간 태스크를 관리하기 위한 Adeos 커널 패치가 필요하다. 현재 Adeos 커널 패치 프로젝트[17]는 모든 Linux 커널 버전에 대하여 제공되지 않고 있다. 따라서 Linux 커널 중 3.10 버전에서 제공되는 3.10.37 Linux 커널 버전 및 Adeos 커널 패치를 채택하여 커널 빌드 및 이식을 진행하였다. 추가적으로 Xenomai RTOS와 관련된 라이브러리를 ARM 프로세서용으로 빌드 후 운영체제 런타임 환경을 구축하였다.

GMC에 이식된 IgH EtherCAT Master Stack은 Generic Driver와 Native Driver의 두 개의 형태로 시스템에 이식할 수 있다. 전자의 경우는 기존의 이더넷 드라이버를 수정하지 않고 EtherCAT 드라이버로 사용되는 형태를 말한다. 이는 별도의 수정 없이 사용하기 때문에 네트워크 드라이버와 관련된 지식 없이도 쉽게 EtherCAT Master를 구성할 수 있다. 그러나 이는 기존의 네트워크 스택을 모두 사용하기 때문에 Native 드라이버에 비해 EtherCAT 네트워크 성능이 낮다. 이와는 반대로 Native 드라이버는 이더넷 드라이버의 별도 수정 과정이 필요하지만 Master가 직접적으로 하드웨어를 접근하여 네트워크 패킷을 처리하기 때문에 높은 실시간 네트워크 성능을 보일 수 있다. 현재 Etherlab[15]에서는 일반적으로 많이 사용하는 Intel社의 e1000, e1000e 및 Realtec社 8139, 8169 칩의 Native Driver 소스 만을 제공하고 있다. 그러나 일반적으로 ARM 프로세서의 임베디드 플랫폼은 판매하는 제조사마다 다른 이더넷 칩과 드라이버가 사용된다. 본 논문의 GMC에 사용된 Freescale社의 imx6q 프로세서에서는 fec 이더넷 칩을 위한 Ethernet Driver가 사용된다. 따라서 3.10.37 리눅스 커널의 fec 이더넷 드라이버를 Etherlab에서 제공하고 있는 Native Driver 개발 가이드에 참고하여 Native Driver 개발을 진행하였다. 세부적으로는 첫 번째, Native Driver의 실시간 성능과 외부 간섭을 없애기 위해 기존 Ethernet Driver에서 사용되는 인터럽트 관련된 API를 제거하였다. 또한 기존의 네트워크 계층을 사용하지 않으므로 netif_*() API와 그 외 불필요한 함수들이 호출되지 않도록 처리하였다. 최종적으로 기존 인터럽트 방식으로 동작되던 Ethernet Driver를 폴링 방식으로 동작되도록 송수신 처리 API를 수정하였다.

3.3 다중 프로세서 기반의 한국형 오픈소스 기반 모션 제어 시스템

본 절에서는 다중 프로세서를 지원하기 위한 KOSMOS 개발 내용에 대해 제시한다. 3.1절에서 제시한 것과 같이 GMC는 범용 Linux 기반의 Xenomai RTOS 환경에서 동작한다. 따라서 현재 Linux 운영체제에서 제공하는 SMP Linux API를 이용하면 별도의 커널 수정 없이 비실시간 프로세스의 병렬 처리가 가능하도록 구현할 수 있다. 또한 Xenomai 태스크는 rt_task_create() Xenomai API를 통해 특정 CPU를 지정하여 실행하도록 할 수 있다. 이를 정리하여 최종적으로 본 논문에서는 PHS(Process Handling Script)를 개발하여 비실시간 태스크를 처리하도록 개발하였으며 KOSMOS 실시간 태스크(Xenomai 태스크)의 성능을 개선하였다. PHS는 KOSMOS 태스크를 제외한 모든 프로세스를 처리하게 되는데 사용자가 원하는 시점에 특정 CPU에서 프로세스가 동작되도록 할 수 있다. 예를 들어 Quad Multi-core인 경우, 사용자가 원하는 태스크를 0번 CPU에서만 동작시키거나 1,3번 CPU 또는 모든 CPU에서 프로세스가 동작되도록 처리할 수 있다. 이러한 기능을 이용하여 KOSMOS 실시간 태스크는 다른 태스크의 간섭 없이 독립적으로 실행되도록 설정할 수 있으며 OPC UA Master, HMI 등과 같이 시스템의 주요 프로세스 또한 별도 관리가 가능하다. 이와 같은 기능은 산업용 PC와 비교하여 비교적 낮은 성능과 제한적인 리소스를 가지는 임베디드 플랫폼인 경우에 더욱 필요하며 시스템 성능을 높이는데 효과적으로 적용할 수 있다.

4. 실험 및 성능 평가

본 논문에서 제안하는 한국형 모션 제어 시스템의 성능 평가를 위해 표 1과 같이 실험 환경을 구성하였다. 표1 좌측은 ARM

표 1 실험 환경
Table 1 Experiment environment

항목	GMC	KOSMOS IDE
프로세서	ARMv7 Quad-core Processor rev 10 v71 @ 1.2GHz	Intel(R) Atom(TM) CPU E3845 @ 1.92GHz
메모리	2GB	4GB
저장장치	Transcend 8GB SD card	2.5“ SATA SSD 3MG-P ATA Device
네트워크 인터페이스	Smsc95xx USB 2.0 Ethernet 1.5.2 IgH Ethercat Master Stack based fec ethercat driver	Realtek RTL8139/810x
운영체제	Linux 3.10.37, Xenomai 2.6.4	Windows 7 Pro 64bit
EtherCAT Slave	Higen EDA 7001 AC Servo Drive	

프로세서 기반의 GMC 하드웨어이며 우측은 4.1절 및 4.2절 실험에서 사용한 KOSMOS IDE와 OPC UA 클라이언트 소프트웨어가 동작되는 사용자 PC를 나타낸다. 모션 제어 시스템의 주요 성능 척도로는 실시간 태스크의 정주기성이 있으며 실시간 태스크의 실행시간 지터 또한 중요 척도로 사용될 수 있다. 실행 시간의 편차가 적을수록 정확한 시스템 구동을 예측할 수 있기 때문이다. 모션 정밀도 또한 주요 성능 척도로 사용될 수 있는데 일반적으로 이는 실시간 이더넷 수신의 정주기성으로 판단할 수 있다. 따라서 이를 통해 4.1절에서는 단일 프로세서 환경에서의 시스템 정주기성을 측정한 결과를 제시하고 4.2절에서는 3.3절에서 제시한 다중 프로세서 환경에서의 시스템 성능 평가를 위해 실행시간에 대해 분석한다.

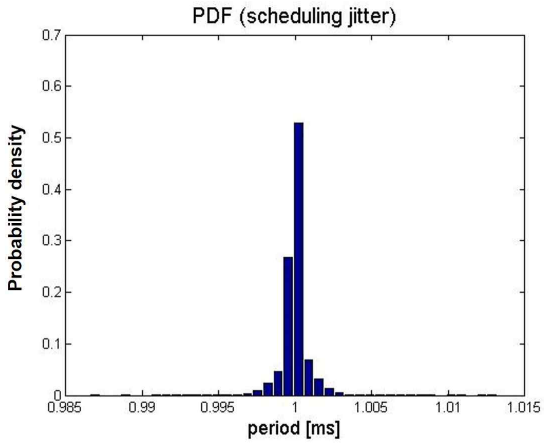
4.1 한국형 모션 제어 시스템 성능 평가

본 논문에서 제안하는 KOSMOS 성능 평가를 위해 우선 단일 프로세서 환경에서의 정주기성을 측정하였다. 실험 환경은 표1과 같으며 KOSMOS 실시간 태스크 정주기성을 확인하기 위해 실시간 태스크의 주기가 시작될 때마다 시스템 시간 값을 측정하는 방식으로 실험을 진행하였다.

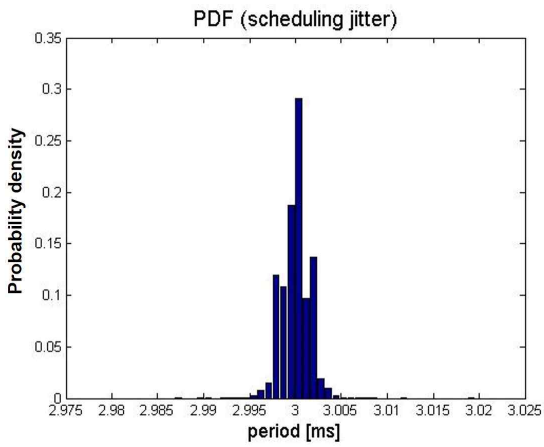
그림 3은 KOSMOS 실시간 태스크를 1ms, 3ms 두 가지 경우의 주기로 측정한 결과이다.

그림 3을 통해 비교적 빠른 1ms에서도 약 0.005ms 이내에 모든 태스크가 주기적으로 실행된 것을 알 수 있으며 60000개의 샘플링 데이터를 측정한 결과 평균 999.99μs와 편차 0.8μs의 결과를 도출해 낼 수 있었다.

두 번째로는 3.2절에서 제시한 것처럼 GMC의 EtherCAT 성능 측정 실험을 진행하였다. 정밀한 모션 제어를 위해서는 비교적 낮은 주기에서도 균일하게 EtherCAT 프레임 수신이 이루어져야 한다. 이를 위해 본 논문에서 EtherCAT을 위한 Native Driver를 개발하였으며 이를 적용하여 GMC의 성능 평가를 진행하였다. 그림 4는 KOSMOS 런타임 구조를 나타낸다. 우선 EtherCAT, PLC 프로그램과 관련된 전역 변수 및 지역 변수 태스크 등을 초기화한다. 그 이후 EtherCAT 프레임 수신, PLC 프로그램 실행 및 EtherCAT 프레임 전송 과정을 주기적으로 반복한다. 따라서 만약 EtherCAT 프레임 수신이 정확한 주기로 실행되지 않는다면 정밀 모션 제어가 불가능하게 됨을 알 수 있다.



(a)주기 1ms



(b)주기 3ms

그림 3 한국형 모션 제어 시스템 정주기성 측정 결과
Fig. 3 Experiment result for the periodicity of KOSMOS

이를 평가하기 위해 1ms 제어 주기의 모션 응용을 작성하여 EtherCAT 프레임 수신 시간을 측정하였다. 실험 환경은 4장에서 제시한 것과 같다. 실험 결과, 그림 5와 같이 13만 개의 샘플링

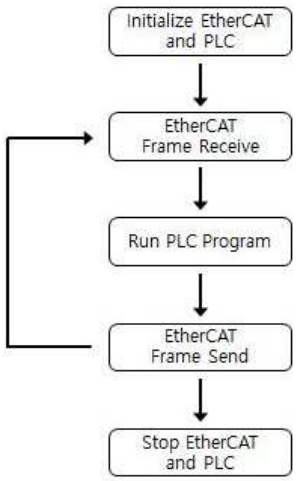


그림 4 한국형 모션 제어 시스템 런타임 구조
Fig. 4 Runtime structure of KOSMOS

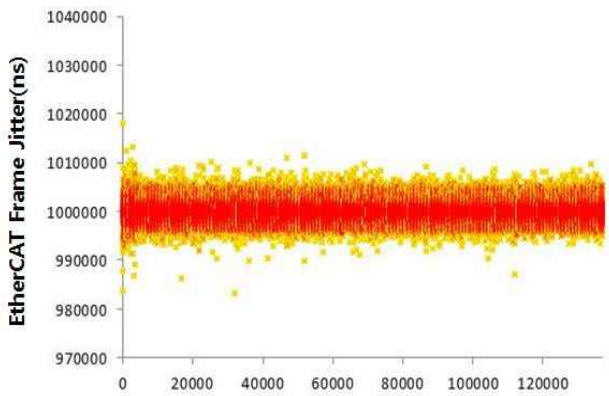


그림 5 EtherCAT Frame 수신 시간 측정
Fig. 5 measurement of EtherCAT frame receive time

데이터에서 평균 999.99μs, 표준 편차 1.27μs임을 확인할 수 있었다. 이를 통해 본 논문에서 개발한 EtherCAT Native 드라이버가 높은 정밀도로 동작됨을 알 수 있으며 비교적 짧은 주기인 1ms에서도 안정된 정밀 모션 제어가 가능함을 알 수 있었다.

4.2 다중 프로세서 기반의 시스템 성능 평가

본 절에서는 다중 프로세서 환경에서의 KOSMOS 시스템 성능 평가를 제시한다. 실험을 진행하기에 앞서 전체 시스템의 오버헤드를 높이고 KOSMOS 실시간 태스크의 간섭 영향을 정밀하게 보기 위해 파일 입출력 및 메모리 접근 응용으로 작성된 2개의 비실시간 태스크와 2개의 실시간 태스크로 시스템을 구성하였다. 추가적으로 외부 OPC UA 클라이언트를 위한 OPC UA Master 태스크 및 KOSMOS 실시간 태스크가 수행된다. 실험 구성1은 KOSMOS 실시간 태스크를 포함한 태스크들이 커널에 의해 할당되는 모든 CPU에서 동작되도록 구현하였다. 실험 구성2는 가상 태스크를 포함한 모든 태스크는 0번과 1번 CPU, OPC UA 태스크는 2번 CPU, KOSMOS 실시간 태스크는 3번 CPU에 지정하여 실험을 진행하였다. 실험 결과는 표 2와 같다. 각 실험에서 7000개의 샘플링 데이터를 측정하였고 총 10번의 실험을 진행하였다. 표2의 결과를 통해 다중 프로세서의 런타임 환경에서 실행 시간의 편차가 적은 것을 확인할 수 있다. 이는 단일 프로세서인 경우 다른 실시간 태스크에 의한 간섭이나 캐시 메모리 오버헤드 등의 영향으로 KOSMOS 실시간 태스크의 오버헤드가 증가함을 알 수 있다. 또한 KOSMOS 실시간 태스크의 실행시간이 긴 경우에는 더욱 크게 영향을 받음을 예측할 수 있다.

표 2 실시간 태스크 실행 시간 측정 결과
Table 2 Experiment result for execution time of real-time task

Index	실험 구성1		실험 구성2	
	Average	Deviation	Average	Deviation
1	550.54μs	114.08μs	388.05μs	2.15μs
2	558.02μs	126.13μs	387.96μs	2.39μs
3	561.46μs	113.75μs	388.39μs	3.68μs
4	562.75μs	114.22μs	388.48μs	4.36μs
5	555.83μs	110.86μs	388.00μs	2.67μs
6	551.24μs	123.47μs	388.23μs	2.66μs
7	562.57μs	125.15μs	388.14μs	3.84μs
8	551.94μs	113.14μs	387.82μs	2.68μs
9	562.51μs	114.66μs	388.16μs	4.89μs
10	543.05μs	125.10μs	388.38μs	4.75μs

4.3 한국형 모션 제어 시스템 실증

본 절에서는 상위 개발된 한국형 모션 제어 시스템의 실증을 위하여 7축 매니퓰레이터 시스템에 적용 운용하였다. 각 축별 위치 제어기, 속도 제어기 및 토크 제어기 등이 모션 API 형태로 운용되어 전체적인 제어기를 구성하여 시스템을 거동한다. 기존 단일 프로세서 환경에서는 직렬적으로 수행되기 때문에 엄밀한 동시/동기 제어가 되지 않는다.

그러나 4.2절에서 개발된 내용과 같이 다중 프로세서 환경으로 확대 적용하여 축별 독립적인 동시/동기 제어가 가능하다. 본 테스트 환경은 그림6과 같이 구성하였다. 7축 매니퓰레이터를 직접 구동 운용하기 위한 모션 제어기로는 GMC를 사용하였으며 KOSMOS IDE를 통해 운용 프로그램 구현 및 조작을 할 수 있다. 실제 사용된 구동 운용 프로그램은 그림6 하단과 같다. 테스트 환경에서는 각 축별 제어기를 별도의 프로세서에 할당하여 구현 운용하였으며, 이를 통해 실질적인 동시/동기 제어가 됨을 확인할 수 있었다. 위치 제어기는 0.5kHz 주기로 수행되며, 속도 제어기는 5kHz, 마지막으로 토크 제어기는 20kHz로 설정되어 운용된다. 각 제어기는 선형제어기로 구성되어 최적의 파라미터로

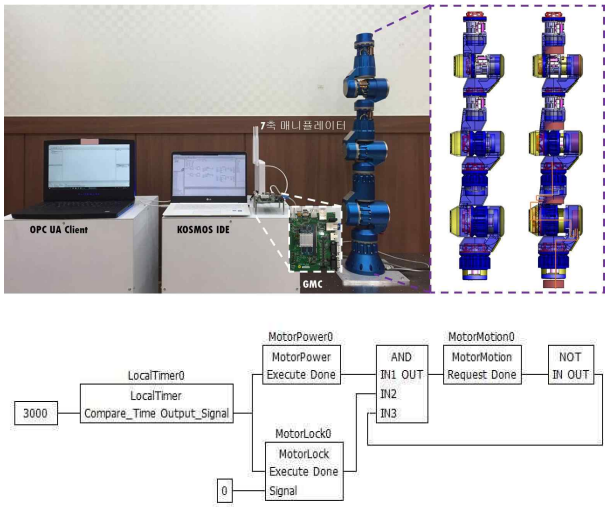


그림 6 2016 로보월드 전시를 위한 KOSMOS 플랫폼
Fig. 6 KOSMOS platform for ROBOTWORLD in 2016

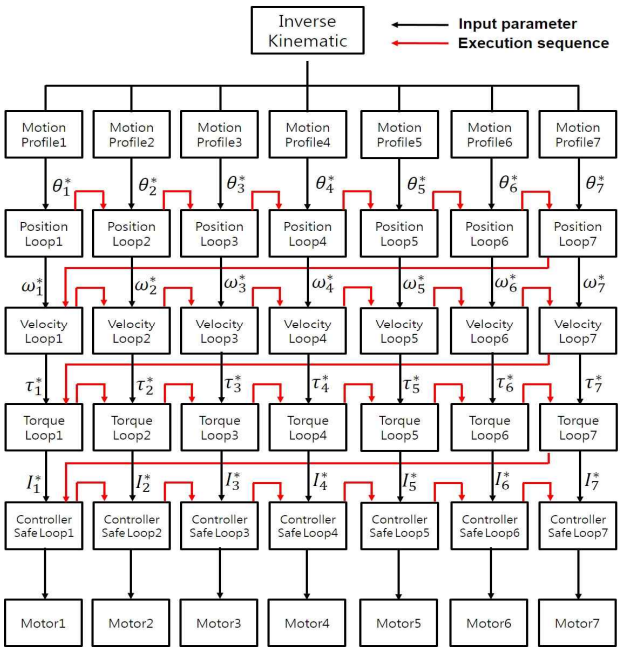


그림 7 단일 프로세서 환경에서의 제어계
Fig. 7 Control system of single processor environment

적절한 튜닝을 한 상태이다. 위와 같이 구성된 제어기를 통해 7축 매니퓰레이터를 운용하여 적용성을 검증하였다. 실제 위치, 속도 및 토크 제어를 계산하는 제어 루프는 그림 7,8과 같다. 그림 7은 단일 프로세서 환경이므로 모든 루프는 순차적으로 실행된다. 그러나 그림 8과 같이 다중 프로세서 환경에서는 물리적으로 4개의 코어를 간섭 없이 할당 및 수행이 가능하므로 최대 4축을 분리 운용할 수 있는 장점이 있다. 단일 프로세스와 다중 프로세

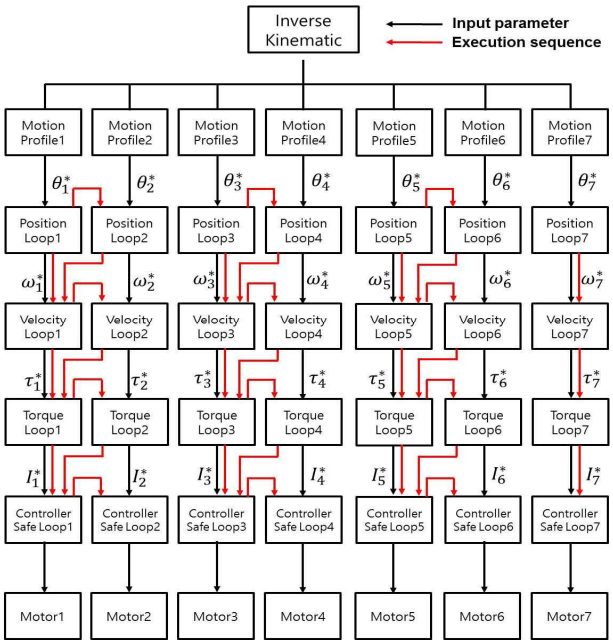


그림 8 다중 프로세서 환경에서의 제어계
Fig. 8 Control system of multi processor environment

스 환경에서 비교 시험을 위한 실제 그림7,8과 같이 각각 구성하여 총 프로그램 실행 시간을 측정하였다. 측정한 결과 단일 프로세서에서는 1426.3μs, 다중 프로세서인 경우 407.5μs로 약 1/3로 실행시간을 단축할 수 있다. 점차적으로 물리적 코어 수가 증가됨에 따라 그 독립 프로세서 환경에서의 적용 가능성은 커지며, 인터럽트 등의 간섭에 따른 수행시간의 감소 및 고속 정주기성의 정시성이 보장될 수 있다.

3. 결 론

본 논문에서는 실시간 이더넷 기반의 한국형 모션 제어 시스템의 구현 및 성능 평가에 대하여 서술한다. 단일 프로세서 및 다중 프로세서 환경에서 KOSMOS 플랫폼을 오픈소스로 이용한 개발 내용과 연구 배경에 대해 제시하였다. 단일 프로세서 환경에서 개발된 시스템의 실시간 성능을 검증하고 이후 다중 프로세서의 성능 평가와 7축 매니퓰레이터로 구성된 모션 제어 시스템의 성능 평가를 진행하였다. 실험 결과, 단일 프로세서 환경에서 1ms, 3ms 주기에서 정밀한 실시간 성능을 가짐을 확인할 수 있었으며 EtherCAT 성능 또한 정밀 모션 제어가 가능함을 확인하였다. 다중 프로세서 기반의 런타임 환경에서는 KOSMOS 실시간 태스크의 실행시간 평균과 표준편차 모두 개선됨을 확인할 수 있었다. 이는 다른 실시간 태스크에 의한 간섭이나 캐시 메모리 오버헤드의 영향을 받지 않아 실행시간이 개선되는 것을 알 수 있었다. 마지막으로 본 논문에서 개발한 GMC 및 다축으로 이루어진 로봇 매니퓰레이터 거동의 적합성을 확인하기 위해 단일 프로세서 및 다중 프로세서 환경에서의 제어 런타임을 분석하고 실험

을 진행하였다. 추가적으로 범용성 측면에서 KOSMOS 플랫폼의 신뢰성과 확장성 확인을 위해 국내 물류 시스템과 도장 로봇 시스템 회사의 공조를 통해 실증을 진행하고 있다. 추가적으로 '스마트 팩토리 기술혁신 컨퍼런스', '사물인터넷 국제전시회', '2016 로보월드'에 참석하여 KOSMOS 플랫폼 실증 결과와 관련하여 홍보를 진행하였다. 향후 연구로는 물류 자동화 공정에서 사용되는 이송 모듈에 적용을 통하여 실증에 대한 연구개발과 소프트웨어 최적화를 진행 후 다른 외산 PLC 제어기와의 비교성능 검증에 대한 연구개발을 계획하고 있다.

감사의 글

본 연구는 산업통상자원부 로봇산업원천기술개발 사업 [No. 10048971, 생산자동화를 위한 표준지향 한국형 오픈소스 모션 플랫폼 기반 범용 모션 제어기 개발]의 지원으로 수행되었음.

References

[1] EtherCAT, "EterCAT Technology Group", Available Online: https://www.ethercat.org/default.htm

[2] Minyoung Sung, Kanghee Kim, et al., "An EtherCAT-based Motor drive for High precision Motion systems," 2011 9th International Conference on Industrial Informatics. IEEE, pp. 163-168, 2011.

[3] Minyoung Sung, Kanghee Kim. "Implementation and Analysis of a Networked Motor Drive using Real-Time Ethernet" KIISE Transactions on Computing Practices, vol 17, no 8, pp 457-462, 2011.

[4] POWERLINK, "Ethernet Powerlink", Available Online: http://www.ethernet-powerlink.org/

[5] V. Stefano, et al., "Real-time Ethernet networks for motion control," Computer standards & interfaces, vol. 33, no. 5, pp. 465-476, 2011.

[6] C. Kim, I. Kim, and T. Kim, "Xenomai-based Embedded Controller for High-Precision, Synchronized Motion Applications," KIISE Transactions on Computing Practices, vol. 21, no. 3, pp. 173-182, 2015.

[7] Xenomai, "Real-time frame for Linux", Available Online: https://xenomai.org/

[8] B. Antonio, et al., "Performance comparison of VxWorks, linux, RTAI and xenomai in a hard real-time application," in Proc. IEEE Real-Time Conference, pp. 1-5, 2007.

[9] B. Jeremy H., and B. Martin, "How fast is fast enough? choosing between Xenomai and Linux for real-time applications," in proc. the 12th Real-Time Linux Workshop (RTLWS'12), pp. 1-17, 2010.

[10] M. D. Marieska, A. I. Kistijantoro, and M. Subair, "Analysis and benchmarking performance of Real Time Patch Linux and Xenomai in serving a real time application," in Proc. IEEE International Conference on Electrical Engineering and Informatics (ICEEI), pp. 1-6, 2011.

[11] J. Koh, and B. Choi, "Real-time performance of real-time mechanisms for rtai and xenomai in various running conditions," International Journal of Control and Automation, vol. 6, no. 1, pp. 235-246, 2013.

[12] Wei, Hongxing, et al., "RT-ROS: A real-time ROS architecture on multi-core processors.", in Future Generation Computer Systems, pp 171-178, 2016.

[13] Sharma, Mridula, Haytham Elmiligi, and Fayez Gebali, "Simulations and performance evaluation of Real-Time Multi-core Systems.", Computer Engineering & Systems (ICCES), pp 214-218, 2014.

[14] Beremiz, "Home Page of Beremiz", Available Online: http://www.beremiz.org/

[15] IgH, "IgH EtherCAT Master for Linux", Available Online: http://www.etherlab.org/en/ethercat/

[16] IEC 61131-3 : the third part (of 10) of the open international standard IEC 61131 for programmable logic controllers.

[17] Adeos, "The Adeos Project", Available Online: http://home.gna.org/adeos/

[18] "Using EtherCAT for Industrial Control Communication", Available Online: http://www2.advantech.com/ePlatform/Motherboards/Whitepaper.aspx?doc_id=bbe5d7b7-4148-4efa-b0ae-c569cbf94c90

저 자 소 개

임 선(Sun Lim)

2002년 고려대학교 전기전자공학과 졸업(학사). 2005년 동대학원 전기공학과 졸업(석사). 2015년 동대학원 전기전자공학과 졸업(박사). 두산기술원 전자기술팀 근무(~2008), 현재 전자부품연구원 지능로보틱스 선임연구원. 관심 분야는 모션 제어 시스템, 오픈 소스 임베디드 시스템, 실시간 시스템, 비선형 제어이론, 적응제어 시스템, 초음파 탐상검사 기법

이 승 용 (Seungyong Lee)

2014년 서울시립대학교 기계정보공학과 졸업(학사). 2016년 서울시립대학교 기계정보공학과 졸업(석사). 현재 전자부품연구원 연구원. 관심 분야는 모션 제어 시스템, 오픈 소스 임베디드 시스템, 실시간 시스템

김 지 현 (Seungyong Lee)

2016년 부경대학교 제어계측공학과 졸업(학사). 현재 전자부품연구원 연구원. 관심 분야는 모션 제어 시스템, 산업 자동화, 임베디드 모션 제어 시스템

정 일 균 (Ilkyun Jung)

1996년 아주대학교 제어공학과 졸업(학사). 1998년 동대학원 제어공학과 졸업(석사). 2004년 INP(National Polytechnic Institute) 컴퓨터공학과 졸업(박사). LG전자 리빙센터 근무(~2006), 현재 전자부품연구원 지능로보틱스 센터장. 관심 분야는 모션 제어 시스템, 오픈 소스 임베디드 시스템, 실시간 시스템