http://dx.doi.org/10.5392/JKCA.2013.13.03.022

# 윈도우즈 상에서 실시간 디바이스 드라이버를 위한 통합 미들웨어

## Integrated Middleware for Real-Time Device Drivers on Windows

조아라, 송창인, 이철훈
충남대학교 컴퓨터공학과

Ah-Ra Jo(ahrajo@cnu.ac.kr), Chang-In Song(jjami1234@cnu.ac.kr),
Cheol-Hoon Lee(clee@cnu.ac.kr)

**요약**

최근 무기체계 산업에서 성능검증을 위해 사용되는 점검장비는 수락시험 시 데이터의 정밀성과 실시간성을 요구한다. 이러한 점검장비는 개발의 편의성을 위해 범용 운영체제인 윈도우즈를 사용하는데 윈도우즈의 경우 실시간성을 제공하지 못하는 문제가 있다. 따라서 본 논문에서는 군용 점검장비의 실시간 통신을 위해 윈도우즈 시스템에 실시간성을 제공하는 RTiK-MP(Real-Time implant Kernel-Multi Processor)을 이용한 통합 미들웨어를 설계 및 구현하였다. 또한 DLL(Dynamic Linking Library)을 사용하여 통합 미들웨어의 API(Application Program Interface)를 사용자에게 제공함으로써 기존의 복잡한 RTiK-MP의 내부 구현을 몰라도 개발이 용이하도록 개발의 편의성을 제공하였다. 본 논문은 군용 점검장비의 통신에 실시간성을 제공할 수 있는 통합 미들웨어를 설계 및 구현한 것으로써 많이 사용되는 TCP/IP LAN과 RS-232를 사용하였다. 마지막으로 성능검증을 위해 CPU 클럭 틱의 수를 반환하는 RDTSC 명령어를 사용하였고, 구현한 통합 미들웨어의 성능을 검증한 결과 TCP/IP 및 RS-232의 경우 각각 1ms 및 4ms 주기에서 오차범위 내에서 정상 동작함을 확인하였다.

■ 중심어 : | 윈도우즈 시스템 | 실시간 통신 | RTiK-MP | TCP/IP LAN | RS-232 |

**Abstract**

For the case of test equipments requiring data accuracy, real-time is highly required in acceptance test for performance evaluation of developed weapons. For convenience' sake, test equipments are usually developed using Windows. However Windows does not support real-time in itself. Thus, in this paper, so as to reduce development time and expenses, we design and implement an integrated middleware for real-time device drivers using RTiK-MP. Using DLL, we also support user API's for the sake of development convenience without details of the complex RTiK-MP structure. We evaluate the performance of the proposed integrated middleware using the RDTSC command which returns the number of CPU clock ticks. The evaluation results show that it operates correctly within error ranges in the periods of 1ms and 4ms for the cases of TCP/IP and RS-232, respectively.

■ keyword : | Windows Systems | Real-Time Communication | RTiK-MP | TCP/IP LAN | RS-232 |

* 본 연구는 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행되었음

접수번호 : #130102-001
접수일자 : 2013년 01월 02일
심사완료일 : 2013년 02월 18일
교신저자 : 이철훈, e-mail : clee@cnu.ac.kr

# I. 서 론

임베디드 시스템이 발달함에 따라 일상생활에서 쓰이는 작은 MP3부터 군사용 미사일, 산업용 로봇 등 많은 분야에서 데이터 수집의 정확성과 실시간 데이터 통신을 요구하고 있다. 특히 유도무기체계에서 사용되는 점검장비는 실시간으로 데이터를 획득하고 평가해야 하기 때문에 실시간성 제공이 필요하다[1]. 이러한 점검장비는 신무기 개발 시 개발된 무기의 성능검증을 위한 수락시험에 필요하며, 사용자에게 개발의 편의성을 제공하기 위해 윈도우즈 운영체제를 사용하고 있다. 하지만 윈도우즈 운영체제에서는 실시간성을 제공하지 못하기 때문에 실시간성이 제공되지 않는 환경에서 이 기종 간의 통신은 주기적인 데이터 전송이 이루어지지 않는 문제가 발생한다. 따라서 윈도우즈 운영체제에 실시간성을 제공하기 위해서는 고가의 구입비용 및 경상 사용료를 가지는 RTX 및 INtime과 같은 상용 솔루션을 사용해야 한다. 그러나 상용 솔루션을 사용할 경우, 개발 및 유지보수 비용이 증가되는 문제를 가지고 있다. 또한 최근 다양한 기능들을 지원하기 위해 점검장비의 구성품이 증가됨에 따라 각 구성품들에 대한 실시간성이 요구되고 있다. 하지만 상용 솔루션을 이용하여 디바이스들에게 실시간성을 제공할 경우 기존의 디바이스 드라이버를 새롭게 구현해야 함으로 추가적인 비용 및 시간을 소모해야 하는 문제점이 있다[2][3].

본 논문에서는 이러한 문제점을 해결하기 위해 점검장비에서 많이 사용되는 통신 디바이스들에게 실시간성을 제공하기 위한 통합 미들웨어의 연구를 진행하였다. 이를 위해 실시간 이식 커널(RTiK-MP)을 이용하여, 점검장비의 디바이스들의 실시간 통신과 사용자에게 편의성을 제공할 목적으로 표준화된 사용자 API를 제공하는 통합 미들웨어를 구현하였다. 이렇게 구현된 통합 미들웨어를 통해 기존의 디바이스 드라이버를 새로 구현할 필요 없이 점검장비에서 많이 사용되고 있는 TCP/IP LAN과 RS-232 등과 같은 디바이스들에게 실시간성을 제공할 수 있음을 검증한다.

본 논문은 2장에서 관련연구로 써드파티 운영체제인 RTX와 INtime에 대해 기술하고, RTiK-MP에 대해 기술하며, 3장에서는 디바이스 드라이버에게 실시간성을 제공하는 통합 미들웨어의 설계 및 구현에 대해 기술한다. 4장에서는 실험환경 및 실험결과를 기술하며, 마지막 5장에서는 결론 및 향후 연구과제에 대해 기술한다.

# II. 관련연구

현재 상용중인 써드파티 운영체제인 RTX와 INtime에 대해 기술하며, 윈도우즈에 실시간성을 제공하기 위해 사용되는 RTiK-MP에 대해 기술한다.

## 2.1 써드파티(Third-Party)운영체제

### 2.1.1 RTX

IntervalZero의 RTX(Real-Time Extension)는 윈도우 XP 또는 윈도우 2k에 고성능 실시간 제어가 가능한 실시간 운영체제의 기능을 부가해주는 확장 소프트웨어로서, 대부분의 개발자들이 익숙하고 편리한 윈도우 환경에서 RTX를 이용하여 기존의 실시간 운영체제의 장점을 최대한 사용할 수 있도록 제공해 준다. 즉, RTX는 순수 실시간 운영체제가 아니라 윈도우의 대중성 및 풍부한 GUI 라이브러리의 장점을 최대한 이용하며 실시간성 등의 제약사항들을 보완해 주는 소프트웨어이다. 국방, 항공, 계측기, 시뮬레이션, 의료 및 로보틱스 등의 많은 산업분야에서 사용되고 있다[4].

### 2.1.2 INtime

인텔사가 개발한 iRMX커널을 사용하고 있는 인텔의 x86/CPU에 특화한 리얼타임 시스템 소프트웨어이다. INtime은 윈도우 XP/윈도우2000을 플랫폼으로 한 제어·계측 시스템에 INtime 소프트웨어를 부가해 시스템 상의 주요한 처리를 INtime 응용프로그램에 고쳐 쓰는 것으로, 윈도우의 유연성은 그대로 유지하면서 보다 높은 신뢰성과 정밀한 리얼타임 퍼포먼스를 보충할 수 있다. INtime은 산업용 컴퓨터(IPC)나, x86 보드 컴퓨터, 범용 PC로, 윈도우(2000/XP/vista)와 협조 동작하는 실시간 운영체제이며, 동일 하드웨어 상에서 윈도우와 동시에 동작하는 실시간 운영체제이다. INtime이 설치 된

윈도우는 2개의 커널이 동작하는 멀티 커널 시스템이 된다. 즉, 2개의 가상 CPU를 사용하는 형태로 동작하게 된다[5].

## 2.2 RTiK-MP(Real-Time implant Kernel-Multi Processor)

### 2.2.1 RTiK-MP의 구성

x86 기반 멀티코어 환경의 윈도우즈에서 실시간성을 제공하는 RTiK-MP의 경우 [그림 1]과 같이 디바이스 드라이버 형태로 이식되어 윈도우즈의 커널 자원 및 하드웨어 자원을 이용할 수 있다. Real-Time HAL Extension에서 x86 멀티코어 하드웨어를 접근 및 제어하고, 윈도우즈 커널에 RTiK Kernel Extension 이 디바이스 드라이버 형태로 확장되어 실시간 쓰레드를 생성하고 관리한다[6].

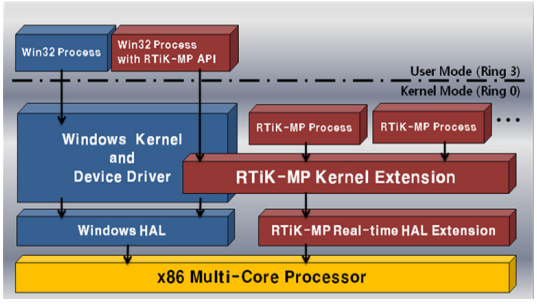


**그림 1. 멀티코어 기반의 실시간 윈도우즈 운영체제 전체 구성도**

### 2.2.2 RTiK-MP의 동작과정

[그림 2]는 RTiK-MP의 인터럽트 처리과정을 나타낸 그림이다. RTiK-MP는 윈도우즈에 실시간성을 제공하기 위하여 Local APIC를 이용해 윈도우즈와는 독립적인 타이머 인터럽트를 발생시킨다. RTiK-MP는 BSP(Boot Strap Processor)가 아닌 AP(Application Processor)의 Local APIC 타이머 인터럽트를 이용하여 윈도우즈와는 독립적인 타이머 인터럽트를 발생시켜 실시간 쓰레드의 주기적인 동작을 보장한다. RTiK-MP는 타이머 인터럽트 처리를 위해 윈도우즈의 지정한 프로세서의 IDT(Interrupt Descriptor Table)에 인터럽트 오브젝트를 등록시킴으로써 인터럽트의 처리가 해당 프로세서에서만 발생되도록 구현되어 있다. 타이머 인터럽트가 발생하기 전에 Local APIC의 타이머 관련 레지스터의 설정과 IDT에 등록되어 있는 인터럽트 벡터를 얻어오게 된다[7-9].

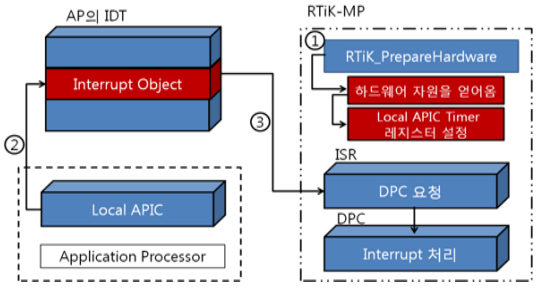


**그림 2. RTiK-MP의 인터럽트 처리과정**

Local APIC 타이머 인터럽트가 발생되면 등록되어 있는 IDT의 인덱스 역할을 하는 Vector를 통해 IDT에 등록되어 있는 인터럽트 핸들러로 분기하게 된다[10][11]. 이때 수행되는 핸들러는 인터럽트 지연시간을 최소화시키기 위해 DPC(Deffered Procedure Call) 큐에 DPC를 등록하게 된다. 인터럽트 핸들러가 완료되고, IRQL(Interrupt Request Level) 값이 Dispatch 레벨이 되면 인터럽트 핸들러에서 등록한 DPC들을 Dispatcher에서 수행하게 된다. 이와 같은 과정을 통해 RTiK-MP는 범용 운영체제인 윈도우즈에서 실시간성을 제공하게 된다. [그림 3]은 이러한 RTiK-MP의 전체적인 동작과정을 나타낸 그림이다[12].

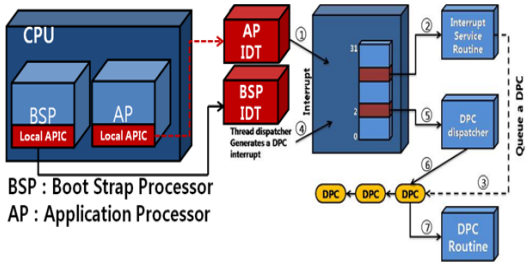


**그림 3. RTiK-MP의 동작방식**

## Ⅲ. 디바이스 드라이버에게 실시간성을 제공하는 통합 미들웨어 설계 및 구현

새롭게 개발된 신무기의 수락시험을 위해 사용되는 점검장비는 실시간으로 데이터를 획득하고 평가하기 위해 이 기종 컴퓨터와 통신 디바이스로 연결되며, 데이터의 실시간 송수신을 필요로 한다. 하지만 점검장비와 이 기종의 컴퓨터 간에 통신 디바이스를 통한 통신은 비동기적으로 발생하기 때문에 실시간 통신이 불가능하다. 이를 위해 실시간 이식 커널(RTiK-MP)을 이용하여, 점검장비의 디바이스들의 실시간 통신과 사용자에게 편의성을 제공할 목적으로 표준화된 사용자 API를 제공하는 통합 미들웨어를 구현하였다. 이에 따라 개발자가 통합 미들웨어를 사용함으로써 기존의 디바이스 드라이버를 새로 구현할 필요 없이 통신에 많이 사용되고 있는 RS-232와 TCP/IP LAN 디바이스 드라이버에게 실시간성을 지원하는 것을 활용할 수 있다.

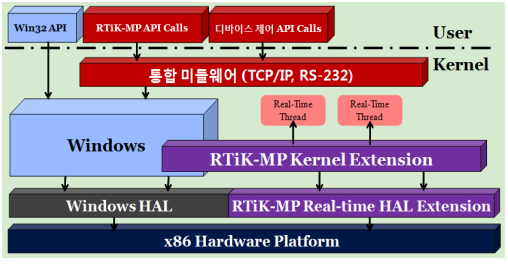


그림 4. 디바이스 드라이버에 실시간성 제공을 위한 통합 미들웨어의 구조도

[그림 4]는 디바이스에게 실시간성을 제공해주는 통합 미들웨어의 구조도이다. x86 기반 멀티코어 환경에서 RTiK-MP가 이식되어 있고, RTiK-MP Real-time HAL Extension에서 커널자원을 접근 및 제어한다. 이를 통해 RTiK-MP가 x86 플랫폼의 Local APIC Register에 접근하여 윈도우즈와는 독립적인 타이머 인터럽트를 발생시키게 된다. 사용자가 통합 미들웨어가 제공하는 API를 이용하여 해당 디바이스 드라이버를 활성화시키면 RTiK-MP Kernel Extension이 활성화된 디바이스 드라이버에게 주기적인 이벤트 시그널을 보내게 되고, 시그널을 받은 디바이스는 윈도우즈의 커널영역에 존재하는 서브루틴들을 수행하게 됨으로써 실시간성을 제공하도록 설계 및 구현하였다.

또한 디바이스의 API를 통합 미들웨어의 API로 매핑시킴으로써 사용자가 복잡한 RTiK-MP의 내부 구현 및 동작과정이나 일반적인 통신 모듈에 대한 이해가 필요하지 않고 쉽게 실시간성을 제공받을 수 있도록 개발의 편의성을 제공하게 된다.

### 3.1 통합 미들웨어의 TCP/IP LAN통신에 실시간성 지원 방법

TCP/IP란 네트워크 전송 프로토콜로, 서로 다른 운영체제를 쓰는 이 기종 컴퓨터 간에도 데이터를 전송할 수 있어 정보 전송을 위한 표준 프로토콜로 쓰이고 있다. TCP는 연결지향 프로토콜로써 데이터 전송 시 발생할 수 있는 패킷의 분실 및 파손을 감지하여 데이터를 재전송하는 Handshacking을 통해 데이터 전송의 신뢰성을 높여준다. 따라서 데이터를 수신하는 호스트에서는 에러 없이 순서대로 데이터를 전송 받을 수 있게 되지만, 이러한 에러 처리에 걸리는 시간이 일정치 않기 때문에 TCP/IP 통신은 실시간성이 제공되지 않는 문제가 있다[13].

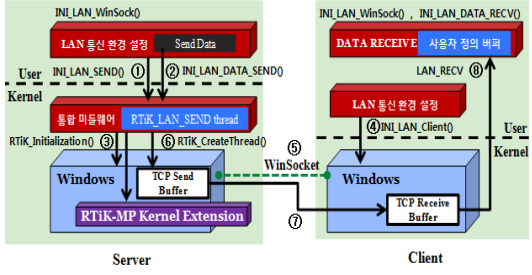


그림 5. 통합 미들웨어를 이용한 TCP/IP 통신 구조

[그림 5]는 통합 미들웨어를 통해 TCP/IP 통신이 이루어지는 과정을 나타낸 그림이다. Server PC와 Client PC가 LAN을 통해 통신을 하게 되고, Server PC에는 RTiK-MP가 이식되어 있다. Server PC에서 Client PC에 데이터를 주기적으로 보내게 되는데, TCP/IP 디바이스의 데이터를 주기적으로 전송하기 위해 윈도우즈 API를 실시간성을 보장하는 RTiK-MP의 실시간 쓰레드와

접목시키고 이를 미들웨어에서 관리하도록 한다. 따라서 사용자는 윈도우즈에서 제공하는 TCP/IP API를 사용하지 않고 통합 미들웨어가 제공하는 API를 사용함으로써 실시간성이 제공되는 TCP/IP 통신을 이용할 수 있다. 사용자가 통합 미들웨어를 통해 Server PC의 LAN통신 환경설정을 마친 뒤, 통합 미들웨어가 제공하는 API를 사용하여 LAN통신의 데이터 송신 코드가 작성된 루틴을 인자로 선택하여 실시간 쓰레드를 생성시킨다.

전송할 데이터를 초기화 시킨 후 Client PC 역시 같은 방법으로 LAN통신 환경 설정을 마치면 윈도우즈 소켓 연결이 이루어지고, 두 PC의 통신 준비가 완료된다. 두 PC 간의 소켓 연결이 이루어지면 실시간 쓰레드가 초기화된 데이터를 사용자가 설정한 주기대로 보내게 되고, Client는 소켓 연결 시 생성된 TCP Receive 버퍼에 데이터를 받아놓고 read명령으로 사용자가 만들어놓은 버퍼에 데이터를 받게 된다.

## 3.2 통합 미들웨어의 RS-232통신에 실시간성 지원 방법

RS-232는 IBM 호환 컴퓨터에서 쓰이는 시리얼 연결로써, 시리얼 포트와 디바이스간 포인트 투 포인트 연결로만 사용이 국한된다. 컴퓨터의 시리얼 포트를 통한 시리얼 통신은 비동기 통신이기 때문에 주기적인 통신이 이루어지지 않으며, 윈도우즈는 시리얼 통신에 대한 실시간성을 보장하지 않는 문제점을 가지고 있다.

윈도우즈 RS-232통신은 [그림 6]과 같이 윈도우즈에서 제공하는 멀티미디어 타이머를 이용한 폴링 방식으로 통신을 제어한다. 통신이 이루어지기 전 Target PC에 RS-232 버퍼가 생성되고, RS-232 포트를 통해 전송된 데이터들은 윈도우즈 멀티미디어 타이머에 의해 15 ms간격으로 RS-232 버퍼로 읽혀지게 된다. 사용자가 정의해 놓은 버퍼로의 수신은 RS-232 버퍼 상태를 수시로 확인하여 읽을 데이터가 있을 경우 데이터를 읽어오는 방식으로 진행된다. 하지만 멀티미디어 타이머의 경우 정확한 주기 안에서의 실행을 보장하지 못하므로 실시간성 제공이 필요하다[14-16].

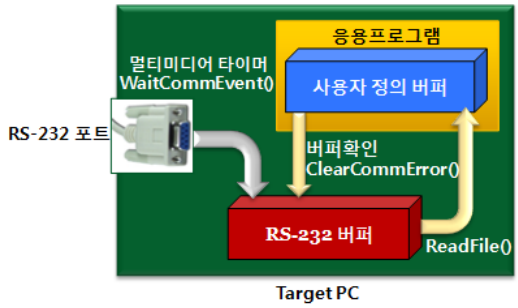


그림 6. RS-232통신의 데이터 수신 과정

실시간성을 제공하는 통합 미들웨어를 이용한 RS-232통신 제어는 [그림 7]과 같이 사용자가 통합 미들웨어를 통해 RS-232통신 포트를 열고, 통합 미들웨어에서 제공하는 API를 사용하여 RS-232통신에 쓰이는 데이터를 설정한다. 사용자가 RS-232통신 에 필요한 데이터 송신 및 수신 코드가 작성 된 루틴을 인자로 선택하여 실시간 쓰레드를 생성하면 미리 입력된 통신 주기대로 데이터의 전송이 이루어진다.

비동기 통신인 RS-232통신의 경우 데이터 송신이 먼저 이루어지면 데이터를 수신하는 Target PC의 RS-232 버퍼에서 버퍼오버플로우 문제가 발생된다. 이를 해결하기 위해 데이터 수신이 먼저 수행 되어야 하되, RS-232 포트에 데이터가 들어온 경우 검사를 통해 수신이 진행 되도록 설계 및 구현을 하였다.

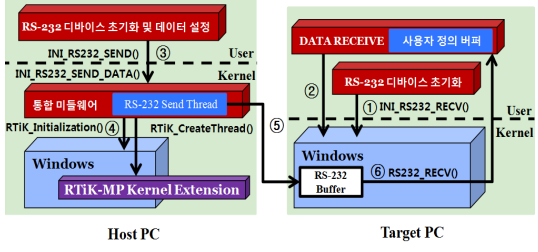


그림 7. 통합 미들웨어를 이용한 RS-232통신 구조

## 3.3 통합 미들웨어 API

기존 RTiK-MP를 이용하여 디바이스에 실시간성을 제공하기 위해서는 RTiK-MP를 초기화하거나 이벤트를 생성하고 동작시키기 위한 구조 및 동작에 대한 이해를 필요로 하기 때문에 많은 코드 작성이 필요하다.

반면 통합 미들웨어를 통해 디바이스에 실시간성을 제공하는 방법은 통합 미들웨어에서 제공되는 API를 통해 코드 사이즈가 작아지며 사용자가 손쉽게 개발할 수 있는 환경이 제공된다. 또한 실시간 통신 시 필요한 API와 구조가 미리 정의되어 있기 때문에 실시간 쓰레드 생성 시 RS-232, TCP/IP 통신 중 사용자가 원하는 통신을 선택하여 동작시킬 수 있다. 통합 미들웨어의 API를 사용자에게 제공하기 위하여 DLL(Dynamic Linking Library)을 사용하였다. DLL은 어플리케이션에서 동적으로 링크하여 사용할 수 있는 라이브러리로써 여러 프로그램이 하나의 DLL파일을 공유하기 때문에 라이브러리가 추가될수록 용량이 커질 수 있는 정적 라이브러리의 단점을 보완하는 링크 라이브러리이다.

[표 1]은 통합 미들웨어가 제공하는 통합 API를 정의한 표이다. 사용자는 표에 정의된 API를 사용하여 통합 미들웨어를 통한 실시간 통신을 실행 시킬 수 있다. RTiK_CreateThread()의 경우 실시간 쓰레드를 생성하는 API로써, RTiK_RS232_SEND나 RTiK_LAN_SEND 중 동작 시키고자 하는 실시간 통신 루틴을 선택하여 인자로 넣게 되면 사용자가 지정해놓은 주기마다 데이터 전송이 이루어지게 된다.

**표 1. 통합 미들웨어가 제공하는 통합 API**

| API | Meaning |
|---|---|
| RTiK_Initialization | RTiK-MP 초기화 |
| RTiK_CreateThread | 동작 시키고자 하는 실시간 통신 쓰레드 생성 |
| RTiK_Enable | RTiK-MP 활성화 |
| RTiK-Disable | RTiK-MP 비활성화 |
| RTiK_Event_clear | RTiK-MP 이벤트 제거 (커널 영역) |
| RTiK_End | 초기화 시에 생성된 이벤트핸들 제거 (유저 영역) |
| INI_LAN_WinSock | 윈도우즈 소켓 생성 |
| END_LAN_WinSock | 윈도우즈 소켓 해제 |
| INI_LAN_DATA_SEND | 전송될 데이터 메모리 할당 |
| INI_LAN_DATA_RECV | 수신될 데이터 메모리 할당 |
| INI_LAN_SEND | Server PC의 소켓 통신 환경 설정 |
| INI_LAN_Client | Client PC의 소켓 통신 환경 설정 |
| LAN_RECV | LAN통신 데이터 수신 |
| INI_RS232_SEND | Host PC의 RS-232통신 환경 설정 |
| INI_RS232_RECV | Target PC의 RS-232통신 환경 설정 |
| INI_RS232_SEND_DATA | RS-232통신에 사용될 데이터 초기화 |
| RS232_RECV | RS-232통신 데이터 수신 |
| RTiK_END_RS232 | RS-232통신 포트 Close |

## Ⅳ. 실험 환경 및 결과

### 4.1 실험 환경 및 실험 방법

디바이스 드라이버에게 실시간성을 제공하기 위한 통합 미들웨어의 성능검증을 위해 멀티프로세서 기반의 윈도우즈 7 환경의 PC에서 실험을 하였다. 이를 위해 실험환경은 [그림 8] 및 [표 2]와 같이 구성하였다.

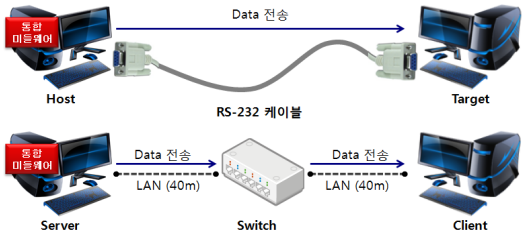


**그림 8. 실험환경**

**표 2. 실험환경**

| RS-232 통신 PC | LAN통신 PC | 프로세서 |
|---|---|---|
| Host | Server | Pentium Dual-Core CPU E6600 @ 3.06GHz |
| Target | Client | Intel Core™ i5-2500 CPU @ 3.30GHz |

Host PC와 Server PC에서 RTiK-MP를 활용한 통합 미들웨어를 사용하여 데이터 전송이 주기적으로 이루어진다. 또한 통합 미들웨어의 성능을 확인하기 위해 클럭틱 카운트 값을 반환하는 RDTSC(Read Time Stamp Counter) 명령어를 이용하여 데이터 송신이 일어나는 주기를 측정했다. 데이터 전송이 발생할 때의 클럭틱 카운트 값을 배열에 저장시키고, 데이터 전송이 모두 이루어졌을 때 배열에 저장해 두었던 클럭틱 카운트의 차이를 계산하는 방법으로 데이터 전송 주기를 확인하였다. 측정된 데이터 전송의 주기 결과와 수신된 데이터 결과는 파일로 저장하여 모두 이상 없이 전송 되었는지 확인하였다. 또한 워크로드는 단순히 while문을 무한 반복하는 윈도우즈의 프로세스로써 윈도우즈의 프로세스가 많아질수록 윈도우즈가 스케줄링을 위한 인터럽트에 의해 인터럽트 지연시간이 길어져도 통신 쓰레드에는 영향을 거의 미치지 않는다는 것을 확인하기 위해 사용하였다.

## 4.2 실험 결과

### 4.2.1 TCP/IP LAN통신 주기측정

[그림 9]는 통신 주기를 RTiK-MP의 최소 주기인 1 ms로 설정하고 수행시킨 통합 미들웨어의 TCP/IP LAN통신 데이터 전송의 주기측정 결과를 그래프로 나타낸 것이다. Excel에서 그래프로 표현할 수 있는 데이터의 최대 개수가 32000개 이므로 32000번의 데이터 전송에 대한 전송주기를 측정하였으며 x축이 나타내는 것은 데이터 전송 횟수이고, y축의 숫자는 주기 값을 의미한다.

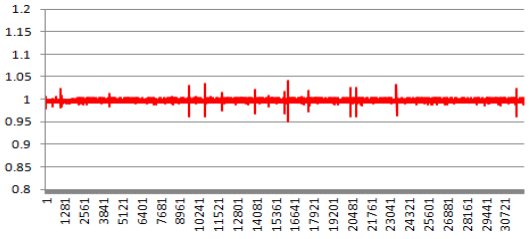


그림 9. 통합 미들웨어의 TCP/IP LAN통신

윈도우즈의 다른 쓰레드가 있을 경우에도 주기적으로 통신이 이루어지는지 확인하기 위해 워크로드를 적용하였으며 [그림 10]은 그 결과를 그래프로 나타낸 그림이다.

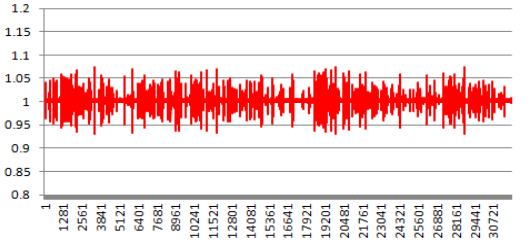


그림 10. 10개의 워크로드 적용 & 통합 미들웨어의 TCP/IP LAN통신 1ms 데이터 전송 주기측정 결과

표 3. 워크로드 적용 1ms 통합 미들웨어 LAN통신

| 전송 주기 | 로드 개수 | 최 대 | 최 소 |
|---|---|---|---|
| 1ms | 로드 없음 | 1.0400ms | 0.9502ms |
| | 로드 5개 | 1.0678ms | 0.8690ms |
| | 로드 10개 | 1.0713ms | 0.9296ms |

[표 3]은 워크로드의 개수에 따른 통합 미들웨어의 TCP/IP LAN통신 1ms 주기측정 결과의 최대, 최소값을 나타내는 표이다. 통합 미들웨어를 통해 TCP/IP LAN 통신 수행 시 워크로드가 없을 경우 약 4%의 오차를 가지며 동작하고, 10개의 워크로드가 적용 될 시 약 7%의 오차로 동작하는 것을 확인할 수 있다. 또한 전송된 데이터의 정확성을 확인하기 위해 수신된 데이터를 순서번호와 함께 파일에 저장하도록 하였다. 실험에 이용된 데이터의 크기는 2KB이며 문자열의 마지막 알파벳을 'a'부터'j'까지 하나씩 증가시키며 전송하는 방법으로 데이터의 순서 및 손실 여부를 확인한 결과 이상 없이 데이터 수신이 이루어지는 것을 확인할 수 있었다.

윈도우즈가 제공하는 큐 타이머와의 비교를 위해 동일한 조건에서 통합 미들웨어의 RTiK-MP와 큐 타이머의 주기를 큐 타이머의 최소 주기인 15ms로 설정하여 TCP/IP LAN통신에 대한 실험을 하였다. [그림 11]과 [그림 12]는 그 결과를 그래프로 나타낸 그림이며 [표 4]와 [표 5]는 각각 적용되는 워크로드의 개수에 따른 큐 타이머와 통합 미들웨어의 전송 주기측정 결과의 최대, 최소값을 나타낸다.

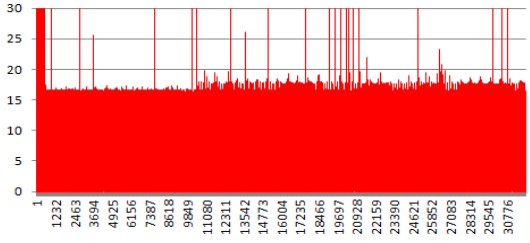


그림 11. 10개의 워크로드 적용 & 큐 타이머의 TCP/IP LAN통신 15ms 데이터 전송 주기측정 결과

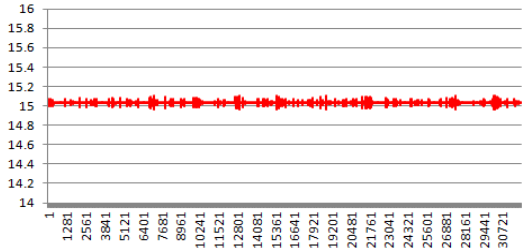


그림 12. 10개의 워크로드 적용 & 통합 미들웨어의 TCP/IP LAN통신 15ms 데이터 전송 주기측정 결과

**표 4. 워크로드 적용 15ms 큐 타이머 LAN통신**

| 전송 주기 | 로드 개수 | 최 대 | 최 소 |
|---|---|---|---|
| 15ms | 로드 없음 | 23.8633ms | 0.0044ms |
| | 로드 5개 | 197.5185ms | 0.0024ms |
| | 로드 10개 | 371.9784ms | 0.0023ms |

**표 5. 워크로드 적용 15ms 통합 미들웨어 LAN통신**

| 전송 주기 | 로드 개수 | 최 대 | 최 소 |
|---|---|---|---|
| 15ms | 로드 없음 | 15.0353ms | 14.9352ms |
| | 로드 5개 | 15.0970ms | 14.9629ms |
| | 로드 10개 | 15.1062ms | 14.9610ms |

큐 타이머를 통한 LAN통신 경우 로드가 걸리지 않은 경우에도 주기를 크게 벗어남을 확인할 수 있는 반면, 통합 미들웨어를 통한 LAN통신 경우 10개의 워크로드가 걸릴 경우 1%이내의 오차로 작동함을 확인할 수 있다.

### 4.2.2 RS-232통신 주기측정

RS-232통신 실험은 1byte 크기의 문자 하나를 통신 데이터로 사용하였고, LAN통신과 동일한 조건으로 실험하였다.

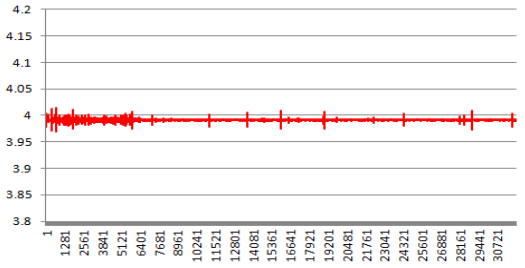


그림 13. 통합 미들웨어의 RS-232통신 4ms 데이터 전송 주기측정 결과

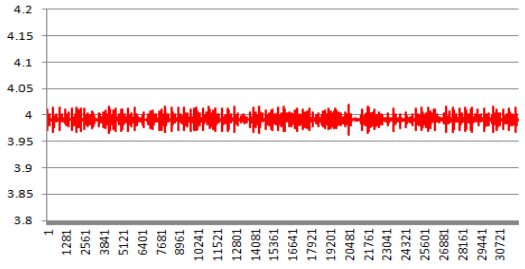


그림 14. 10개의 워크로드 적용 & 통합 미들웨어의 RS-232 통신 4ms 데이터 전송 주기측정 결과

[그림 13]은 통합 미들웨어의 통신 주기를 RS-232의 최소주기인 4ms로 설정하여 RS-232통신을 수행한 결과이며, [그림 14]는 워크로드를 적용했을 시 전송 주기 측정 결과를 나타낸다.

[표 6]은 워크로드 개수에 따른 통합 미들웨어의 전송 주기측정 결과를 나타낸 것이며, 통합 미들웨어를 통한 RS-232통신의 데이터 전송주기는 큐 타이머와는 달리 워크로드에 많은 영향을 받지 않고 1%이내의 오차로 작동함을 확인할 수 있다.

**표 6. 워크로드 적용 4ms 통합 미들웨어 RS-232통신**

| 전송 주기 | 로드 개수 | 최 대 | 최 소 |
|---|---|---|---|
| 4ms | 로드 없음 | 4.0193ms | 3.9629ms |
| | 로드 5개 | 4.0133ms | 3.9690ms |
| | 로드 10개 | 4.0134ms | 3.9687ms |

RS-232통신 주기측정도 마찬가지로 윈도우즈에서 제공하는 큐 타이머와의 차이를 확인하기 위해 큐 타이머와 통합 미들웨어의 통신 주기를 큐 타이머의 최소주기인 15ms로 설정하여 데이터 전송이 이루어지는 주기를 측정하였다.

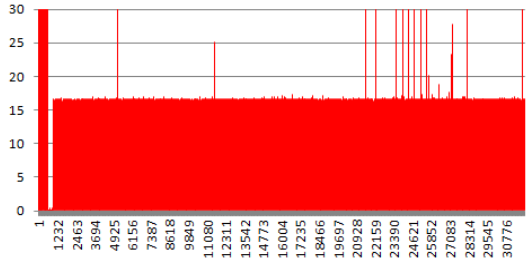


그림 15. 10개의 워크로드 적용 & 큐 타이머의 RS-232 통신 15ms 데이터 전송 주기측정 결과

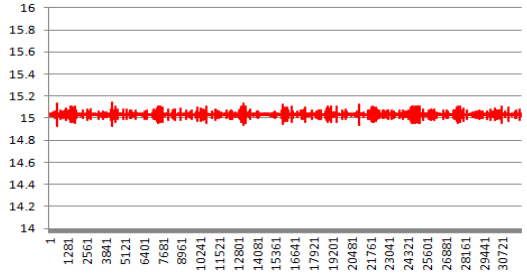


그림 16. 10개의 워크로드 적용 & 통합 미들웨어의 RS-232통신 15ms 데이터 전송 주기측정 결과

[그림 15]와 [그림 16]은 각각 15ms로 주기 설정이 된 큐 타이머와 통합 미들웨어의 RS-232통신전송 주기측정 결과이며, [표 7]과 [표 8]는 각각 워크로드의 개수에 따른 큐 타이머와 통합 미들웨어의 전송 주기측정 결과의 최대, 최소값을 나타낸다.

**표 7. 워크로드 적용 15ms 큐 타이머 RS-232통신**

| 전송주기 | 로드개수 | 최 대 | 최 소 |
|---|---|---|---|
| 15ms | 로드 없음 | 17.7210ms | 0.0155ms |
| | 로드 5개 | 255.1446ms | 0.0144ms |
| | 로드 10개 | 265.9099ms | 0.0120ms |

**표 8. 워크로드 적용 15ms 통합 미들웨어 RS-232통신**

| 전송주기 | 로드개수 | 최 대 | 최 소 |
|---|---|---|---|
| 15ms | 로드 없음 | 15.0435ms | 14.9202ms |
| | 로드 5개 | 15.1093ms | 14.9561ms |
| | 로드 10개 | 15.1417ms | 14.9252ms |

큐 타이머를 통한 RS-232통신 경우 로드가 걸리지 않은 경우에도 주기를 크게 벗어남을 확인할 수 있는 반면 통합 미들웨어를 통한 RS-232통신 경우 10개의 워크로드가 걸릴 경우 1%이내의 오차로 작동함을 확인할 수 있다.

다음과 같은 실험에서 RTiK-MP가 이식된 통합 미들웨어를 사용하여 통신을 수행함으로써 윈도우즈에서 동작하는 프로세스의 수에 영향을 받지 않고, 설정된 주기를 지키며 데이터 전송이 이루어지는 것을 확인할 수 있다.

## V. 결론 및 향후 연구과제

최근 군 장비 및 항공분야에서의 통신 네트워크에 대한 기술이 비약적으로 발달됨에 따라 데이터 수집의 정확성 및 실시간 데이터 통신이 요구되고 있는 추세이다. 또한 새롭게 개발되는 신무기의 성능검증을 위한 점검장비의 경우 개발의 편의성을 위해 윈도우즈 운영체제를 사용하고 있다. 하지만 윈도우즈 운영체제의 경우 실시간성을 지원하지 않기 때문에 점검장비를 구성하는 디바이스 드라이버에게 실시간성을 제공하기 위해서는 RTX 및 INtime과 같은 상용 솔루션을 사용하여 실시간성을 제공해야 한다. 이러한 상용 솔루션을 사용할 경우 고가의 구입비용 및 경상사용료로 인해 개발 비용이 증가되며, 기존의 디바이스 드라이버를 새롭게 구현해야 하므로 개발비용 및 개발에 소요되는 시간이 증가되는 문제가 발생한다. 따라서 기존에 개발된 RTiK-MP를 이용하여 통신 디바이스들을 제어하며 각 디바이스 드라이버에게 실시간성을 제공하는 방안 연구와 더불어 개발 편의성을 제공하기 위한 연구가 필요하다.

본 논문에서는 통합 미들웨어를 구현하여 일반적인 통신에 많이 사용되는 RS-232와 TCP/IP LAN을 제어하도록 하였고 실시간성을 제공하기 위해 RTiK-MP를 이용해 주기적인 통신이 이루어지도록 하였다. 또한 통합 미들웨어에서 제공해주는 API를 작성하여 사용자가 RTiK-MP의 동작과정 및 통신 모듈에 대한 이해가 필요하지 않고 쉽게 실시간성을 제공 받을 수 있도록 개발의 편의성을 제공하였다. 통합 미들웨어의 성능검증을 위해 클럭틱 카운트 값을 반환하는 RDTSC 명령어를 사용하여 데이터 전송의 주기를 측정하였고 데이터 수신의 정확성을 확인하기 위해 수신한 데이터는 파일로 저장한 뒤 확인하였다.

향후 연구과제로는 무기체계에 자주 사용되는 통신 방식인 MIL-STD-1553B[2]를 본 통합 미들웨어에 포함시킴과 동시에 군사용 점검장비에 국한되지 않고 더욱 많은 분야에 본 연구를 적용시키기 위해 다양한 환경에서의 실험 및 성능분석이 필요하다. 또한 현재는 일대일 통신으로 구현이 되어있지만 일대다 통신에 대한 구현 및 테스트가 필요하다.

## 참 고 문 헌

[1] C. H. Koo and H. H. Lee, “Distributed simulator design by using of SimNetwork to overcome speed limit on GenSim,” Recent Advances in Space Technologies (RAST), 2011 5th International Conference on, pp.430-435, 2011(6).

[2] 이승훈, 조아라, 김효중, 조한무, 박영수, 이철훈,

"윈도우즈 시스템 상에서의 군용 점검장비를 위한 실시간 통신", 한국차세대컴퓨팅학회논문지, 제8권, 제4호, pp.47-57, 2012(8).
[3] 주민규, 이진욱, 장철수, 김성훈, 이철훈, "윈도우 유저 레벨 로봇 컴포넌트에 실시간성 지원 방법", 한국콘텐츠학회논문지, 제11권, 제7호, pp.51-59, 2011(7).
[4] http://www.intervalzero.com
[5] http://www.tenasys.com
[6] 주민규, 이진욱, 김종진, 조한무, 박영수, 이철훈, "x86 기반의 윈도우즈 상에서 실시간성 지원 방법", 한국차세대컴퓨팅학회논문지, 제11권, 제4호, pp.47-58, 2011(8).
[7] Intel, *"Intel 64 and IA-32 Architectures Software Developer's Manual Volume 1 : Basic Architecture,"* 2012.
[8] Intel, *"Intel 64 and IA-32 Architectures Software Developer's Manual Volume 3 : System Programming Guide,"* 2012.
[9] Intel, *"Intel 64 and IA-32 Architectures Software Developer's Manual Vol.2 : Instruction Set Reference,"* Intel, 2012.
[10] Intel, *"Intel 64 Architectures x2APIC Specification,"* Intel, Intel 2008.
[11] D. A. Godse and A. P. Godse, *"Microprosessors,"* Technical Publications Pune, pp.432-472, 2007.
[12] 송창인, 이승훈, 주민규, 이철훈, "멀티프로세서 윈도우즈 상에서 실시간성 지원", 한국콘텐츠학회논문지, 제12권, 제6호, pp.68-77, 2012(6).
[13] 윤성우, *TCP/IP 소켓 프로그래밍*, ㈜프리렉, 2003.
[14] O. Bailey, *"Embedded systems : desktop integration,"* Wordware Publishing, Inc., pp.34-56, 2005.
[15] W. A. Ruh, F. X. Maginnis, and W. J. Brown, Enterprise Application Integration: A Wiley Tech Brief, Robert Ipsen, pp.39-52, 2002.
[16] M. Wiberg, EThe Interaction Society: Practice, Theories And Supportive Technologies, Information Science Publishing, pp.215-249, 2005.

## 저 자 소 개

**조 아 라(Ah-Ra Jo)** 준회원

- 2012년 2월 : 충남대학교 컴퓨터공학과(공학사)
- 2012년 2월 ~ 현재 : 충남대학교 컴퓨터 공학과 석사과정 재학

<관심분야> : 실시간 운영체제, 임베디드 시스템, 실시간 통신

**송 창 인(Chang-In Song)** 준회원

- 2010년 8월 : 충남대학교 컴퓨터공학과(공학사)
- 2010년 8월 ~ 현재 : 충남대학교 컴퓨터 공학과 석사과정 재학

<관심분야> : 실시간 운영체제, 내장형 시스템, 로봇 미들웨어

**이 철 훈(Cheol-Hoon Lee)** 정회원

- 1983년 2월 : 서울대학교 전자공학과(공학사)
- 1988년 2월 : 한국과학기술원 전기 및 전자공학과(공학석사)
- 1992년 2월 : 한국과학기술원 전기 및 전자공학과(공학박사)
- 1983년 3월 ~ 1986년 2월 : 삼성전자 컴퓨터 사업부 연구원
- 1992년 3월 ~ 1994년 2월 : 삼성전자 컴퓨터 사업부 선임연구원
- 1994년 2월 ~ 1995년 2월 : Univ. of Michigan 객원 연구원
- 1995년 2월 ~ 현재 : 충남대학교 컴퓨터공학과 교수
- 2004년 2월 ~ 2005년 2월 : Univ. of Michigan 초빙 연구원

<관심분야> : 실시간시스템, 운영체제, 고장허용 컴퓨팅, 로봇 미들웨어