IEEE 1394 네트워에서 실시간성 보장을 위한 디바이스 드라이버 소프트웨어 구조 설계 및 구현

박 동 환**, 임 효 상*, 강 순 주*

A Design and Implementation of Device Driver Architecture of IEEE 1394 Network Adaptor for Guaranteeing Real-Time Characteristics

Dong-Hwan Park**, Hyo-Sang Lim*, Soon-Ju Kang*

요 약

핫 플러깅과 네트워 자동 재구성, 등시성 전송 기능을 지원하는 IEEE1394는 멀티미디어 디지털 홈 네트워의 표준이 되었다. 특히 최근 IEEE 1394 프로토콜이 홈 씨어터 서비스와 같은 QoS 보장형 멀티미디어 네트워, 실시간 통신 기능을 가지는 디지털 계측 제어 프로토콜과 연동되는 환경에서 백본 네트워 프로토콜, 혹은 실시간 코바(CORBA) 와 같은 실시간성을 지원 해야 하는 미들웨어의 물리계층 프로토콜 등에 사용되면서 네트워 디바이스 드라이버 수준에서의 실시간성 보장이 요구되고 있다. 실시간성을 보장하기 위해 IEEE 1394 네트워 디바이스 드라이버는 우선 순위 기반의 패킷 처리 기능과 1394의 등시성 통신의 주기에 기반한 등시성 버퍼관리 기능의 지원이 필요하다. 그러나 기존 상용 OS의 네트워 디바이스 드라이버는 등시성 전송과 같은 IEEE 1394의 특성을 제대로 반영하지 못하며 실시간 통신을 지원하지 않는다. 본 논문에서는 IEEE 1394 디바이스 드라이버 수준에서 실시간 전송을 보장하기 위한 네트워 디바이스 드라이버의 구조를 제안한다.

ABSTRACT

The IEEE 1394 protocol is a de facto standard in multimedia digital home network. It supports several advanced features such as hot plugging, dynamic network reconfiguration, isochronous transmission and so on. Since the IEEE 1394 was adapted in the field of multimedia service with QoS guarantee, back bone network protocol with real-time digital instrumentation and control sub networks, and physical layer protocol for real-time middleware such as real-time CORBA, the additional real-time features has been required in device driver implementation because of the necessity of the predictability enhancement. To guarantee the real-time features, the device driver of the IEEE 1394 should support the priority based packet processing and also need a isochronous buffer management mechanism to deal with the periodic isochronous communication. In this paper, we proposed a new software architecture of the IEEE 1394 device driver for supporting the real-time characteristics such as priority based packet processing, priority based scheduling and so on.

Ⅰ. 서 론

가전기기가 지능화 되어가고 있으며 가정 자동화, 인터넷 셋탑박스, 인터넷 TV를 통한 인터넷 접속 등의 요구가 증가되어 홈 네트워 구성을 위한 다양한 프로토콜과 미들웨어 표준[1,2]이 등장하고 있다. 이 중에서 네트워 비전문가인 일반인들도 가전기기

* 경북대학교 전자전기컴퓨터공학과 ** 한국전자통신연구원

논문번호 : 010367-1203, 접수일자 : 2001년 12월 3일

를 쉽게 홈 네트웍에 연결할 수 있어야 하고 사용자의 개입 없이 자동으로 네트웍이 재구성되어야 한다는 홈 네트웍으로서의 요구와 디지털 캠코더, 디지털 VCR과 같은 영상 기기 VOD (Video On Demand)와 같은 높은 대역폭과 QoS (Quality of Service)보장이 필요한 서비스를 지원하기 위한 멀티미디어 네트웍으로서의 요구를 동시에 만족하는 IEEE 1394가 디지털 가전 기기 간의 네트웍 표준으로 자리잡고 있다[3-7]. 뿐만 아니라 IEEE 1394가 완고한 통신 기능과 우선 순위 처리 기능을 가지는 디지털 제어 계측 프로토콜(예, LonTalk[11], FieldBus[9-10])과 연동되어 백본 네트웍으로 사용되기 시작하면서 1394 네트웍에서의 실시간성 보장이 요구되고 있으며, 특히 최근에는 실시간 코바(CORBA)[12,13]와 같은 실시간 미들 웨어상에서 물리계층 프로토콜로 IEEE1394가 사용됨으로 인해 미들웨어 상에서의 우선순위가 디바이스 드라이버 수준에서 유지될 수 있어야 실시간 미들웨어의 특성을 살릴 수 있게 된다[21].

고속 네트웍에 연결된 가전 기기나 워크스테이션에서의 실시간성 보장을 위해서는 네트웍을 통해 전송되는 패킷을 처리하는 네트웍 디바이스 드라이버를 포함하는 시스템 소프트웨어에서 각 패킷의 특성에 따라 패킷 처리의 우선 순위를 정하고 이에 기반한 데이터의 처리가 이루어져야 한다. 그러나, 기존 운영체제에서 사용되고 있는 네트웍 디바이스 드라이버는 FIFO 큐를 사용하여 패킷의 처리를 도달 순서에 따라 처리함으로써 각 패킷이 처리 되는데 걸리는 시간을 예측할 수 없게 하는 구조를 가지고 있어 실시간성을 요구하는 응용에 사용할 수 없으며 이더넷과 같은 비동기 통신을 기반으로 하는 구조를 가지고 있어 IEEE 1394가 제공하는 등시성 (Isochronous) 전송 방식의 특성을 고려하지 않았다[14].

본 논문에서는 실시간성을 보장하며 동시에 IEEE 1394 네트웍의 특성을 반영한 네트웍 디바이스 드라이버의 구조를 제안한다. 제안한 구조에서는 패킷 처리에 우선 순위 큐를 이용함으로써 낮은 우선 순위의 패킷 처리에 의해 높은 우선 순위의 패킷 처리가 무한히 지연되지 않도록 하여 패킷이 처리되는 데 소요되는 시간을 예측할 수 있게 하며, 네트웍을 통한 패킷 수신 시 처리해야 할 작업을 인터럽트 서비스 루틴 내에서 처리해야 할 내용과 그렇지 않은 부분으로 세분화하여 인터럽트 내에서 패킷을 처리함으로써 발생하는 인터럽트 지연 시간(Interrupt Latency)을 최소화할 수 있는 구조로 설계하였다. 그리고 IEEE 1394의 등시성 전송 방식을 처리하는 태스크들은 주기를 가지게 되며 각 태스크의 데드라인 만족시키기 위해 주기에 따른 동적 우선 순위 스케쥴링 방식인 레이트 모노토닉 스케쥴링을 적용하였다.

본 논문에서는 서론에 이어 2장에서는 IEEE 1394의 특성과 이를 기반으로 한 홈 네트웍 및 관련 연구에 대해 알아보고, 3장에서는 실시간성 보장을 위한 요구와 서비스 시나리오 그리고 네트웍 디바이스 드라이버 비동기 링크 유닛, 등시성 링크 유닛, 트랜잭션 유닛의 설계에 대해 설명하고 4장에서는 제안한 네트웍 디바이스 드라이버 구현과 이를 이용한 제안한 구조의 성능을 평가한 결과를 보여주고 5장에서 결론을 맺도록 하겠다.

Ⅱ. 기본 개념 및 관련 연구

1. IEEE 1394 특성

IEEE 1394[3-7]는 1980년대 중반 애플 컴퓨터 사에서 FireWire란 이름으로 개발되기 시작하였고 1995년 IEEE 위원회에서 표준으로 상정되어 IEEE 1394-1995 표준안이 되었다. IEEE 1394의 설계 목표는 다양한 디바이스들을 고성능 직렬 버스에 접속시키는 것이었다. 이를 위해서 사용의 편이성, 낮은 개발비, 고속 전송 지원, 성능 확장성, 등시성 응용 지원, 많은 메모리 주소 공간 지원, 호스트 시스템에 독립적인 동작 등의 설계 개념에 바탕을 두고 개발 되었으며 그 특성을 요약하면 다음과 같다.

(1) IEEE 1394는 100 Mbps, 200 Mbps 및 400 Mbps의 전송 속도를 동시에 지원한다. 이를 위해서 버스 초기화 시에 각 디바이스의 최고 속도를 알아내 최소의 전송속도가 되도록 버스의 대역폭을 조절한다. 따라서 전송 속도가 다른 디바이스들이 하나의 버스를 통해 통신할 수 있다.

(2) IEEE 1394에서 각 디바이스는 버스가 동작중에도 접속 및 단절이 가능한 핫 플러깅(Hot Plugging) 기능을 지원한다. 버스 동작 중에 디바이스가 접속 또는 단절되면 버스는 자동으로 각 디바이스에 새로운 주소를 할당하고 이를 시스템 소프트웨어에 알린다. 이런 과정을 통해서 새로운 디바이스가 서비스 중에 추가되거나 제거되어도 디바이스 동작에 끊임 없이 서비스가 가능하다.

(3) IEEE 1394에서는 전송 여부를 확인할 수 있는 신뢰할 수 있는(reliable) 데이터 전송 방식과 등

시성 전송 방식 두 가지를 동시에 지원한다. 신뢰할 수 있는 데이터 전송을 지원하는 비동기 전송 방식은 에러 처리 기능을 가지고 있어 데이터의 손실을 방지하며 등시성 전송 방식에서는 실시간 전송에 필요한 QoS를 보장한다. 이 전송 방식은 전송 전에 필요한 대역폭을 등시성 자원 관리자(Isochronous Resource Manager)로부터 할당 받아 주어진 대역폭으로 정확한 시간에 데이터를 전송하여 QoS를 보장하지만 실시간 전송을 위해서 에러 처리 기능을 지원하지 않는다. 이 등시성 전송은 비디오 및 오디오 데이터 전달에 주로 사용된다.

(4) IEEE 1394에서는 하나의 네트워크에 1024개의 버스를 연결할 수 있으며, 각 버스마다 64개의 디바이스를 연결할 수 있다.

(5) IEEE 1394에서는 데이터의 전송이 호스트 컴퓨터 시스템의 개입 없이 이루어지기 위해서 피어-투-피어(peer-to-peer) 트랜잭션만을 지원한다. 각 디바이스는 호스트 컴퓨터 시스템의 조정 없이 버스에 데이터를 전달할 수 있다. 따라서 컴퓨터 시스템이 없는 디바이스를 붙여서 사용 가능하므로 가정용 홈 네트워크에 적합하다.

(6) 등시성 전송을 위해서 할당된 대역폭을 보장할 수 있는 공정한 중재(fair arbitration) 알고리즘을 이용한다. 이 알고리즘을 통해 IEEE 1394에서는 비동기 전송도 균일한 대역폭을 할당 받을 수 있다.

(7) IEEE 1394 케이블에 파워 라인도 같이 있어, 각 디바이스는 파워 소스 또는 싱크로서 역할을 할 수 있다.

2. IEEE 1394 프로토콜 스택

IEEE 1394는 계층 구조를 이용하여 프로토콜을 구현하고 있다. 그림 1은 IEEE 1394의 계층을 나타낸다. 기본적으로 IEEE 1394는 트랜잭션 계층, 링크 계층, 물리 계층의 세 개 계층으로 구성되어 있으며 이 외에 버스 관리를 위해 직렬 버스 관리 계층을 가지고 있어 각 계층과 연관되어 있다. 직렬 버스 관리 계층은 버스 초기화와 관리를 담당하고 버스 관리자(Bus Manager), 등시성 자원 관리자(Isochronous Resource Manager)와 노드 제어기(Node Controller)로 구분된다. 버스 관리자는 소프트웨어와 연동하는 부분이고 등시성 자원 관리자는 버스내의 등시성 채널과 등시성 전송을 위한 대역폭을 관리하는 기능을 담당하고 노드 제어기는 버스 초기화 시에 노드 트리 결정, 노드 주소 결정, 대역폭 할당 등의 일을 담당한다.

트랜잭션 계층은 비동기 전송을 발생 시키는 계층으로 CSR(Control Status Register)[7]의 READ, WRITE, LOCK을 지원한다. 그러나 등시성 송수신은 트랜잭션 계층을 거치지 않고 응용에서 직접 링크 계층을 통해 이루어진다. 링크 계층은 트랜잭션 계층에서 생성된 트랜잭션을 해석하여 패킷으로 변경하여 전달하고 전달 받는 역할을 하며 또한 수신된 비동기 패킷에 대해 승인 패킷(Acknowledge packet)을 전송하여 신뢰할 수 있는 데이터 전송을 지원한다. 물리 계층은 데이터 전송에 필요한 전기적인 특성을 정의하고 있으며 버스 중재 (Bus Arbitration) 기능을 구현하여 버스를 통해서 하나의 노드만이 데이터를 전송할 수 있게 보장한다.

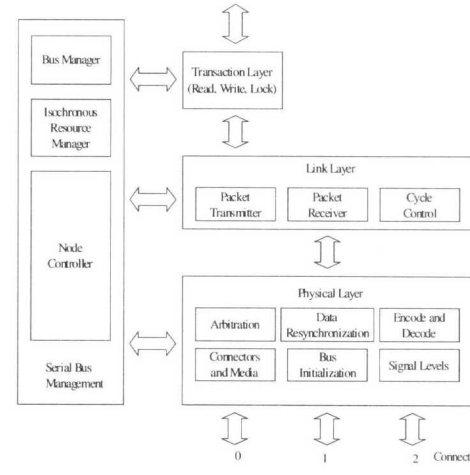


그림 1. IEEE 1394 프로토콜 계층

3. 실시간성 보장 디바이스 드라이버의 필요성

IEEE 1394를 통해 전달될 수 있는 정보로는 실시간성 보장을 필요로 하지 않는 일반 데이터 뿐 아니라 비디오, 오디오와 같은 멀티미디어 스트림 데이터나 디지털 계측 장비를 통한 연속적인 센서 신호 또는 로봇 제어 명령과 같은 실시간성이 보장되어야 하는 데이터를 동시에 포함하고 있다. 이런 실시간성이 보장되어야 하는 데이터를 처리하기 위해서는 네트워크로 송수신되는 모든 패킷 처리를 담당하는 네트워크 디바이스 드라이버에서의 실시간성 보장이 필요하다. 또한 실시간 코바(CORBA)와 같은 실시간 미들웨어를 IEEE1394를 백본 네트워크로하는 홈 네트워크에 적용하여 다양한 이기종 프로토콜을 지원하는 홈 디바이스들간의 유연한 연동과 실시간성 보장을 위한 홈 네트워크 서비스들이 계획되어 지고 있다. 그림 2는 이러한 IEEE실시간 미들웨어를 적용시킨 홈 네트워크 개념도[20]이다.

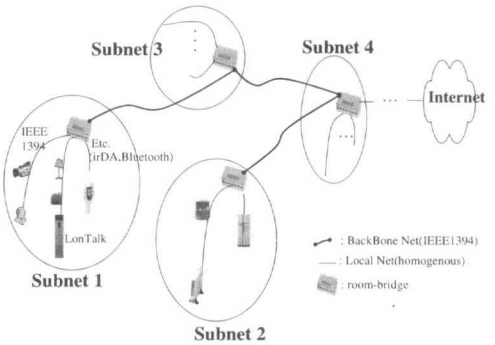


그림 2. IEEE1394를 백본 프로토콜로 사용하는 홈 네트워크 개념도

위 그림에서 보듯이 IEEE1394를 백본 네트워크로 사용하고 있기 때문에 우선 순위 패킷 처리 능력을 가지고 있는 LonTalk[11] 네트워크의 패킷들에 대한 우선순위를 IEEE1394 네트워크에서 유지시켜야만 실시간성을 요구하는 홈 오토메이션 서비스들이 가능할 수 있게 된다. 즉, 기존 네트워크에서 연구된 실시간성 보장을 위한 응용들을 IEEE1394에서도 그대로 적용하기 위해서는 1394 네트워크 디바이스 드라이버에서의 실시간성 보장이 필수적이다.

네트워크을 사용하는 다수의 응용이 네트워크 디바이스 드라이버를 통해 각각의 프로토콜에 따라 동시에 패킷을 전송하고 수신하기 때문에 네트워크 자원을 공유하게 되어 서로의 동작에 영향을 미치게 된다. 그러므로 공유되는 네트워크 자원을 관리하는 네트워크 디바이스 드라이버 차원에서의 실시간성 지원이 보장되지 않는다면 네트워크 디바이스 드라이버에서의 패킷 처리 및 전달 시간의 예측이 불가능하게 되며 이를 이용하여 통신하는 응용에서의 명령 처리 시간 예측 또한 불가능하게 되므로 네트워크 디바이스 드라이버에서의 실시간성 보장은 응용에서의 실시간성 보장을 위한 필요 조건이다.

이런 이유로 네트워크에서의 실시간성 보장을 위한 많은 연구가 이루어 지고 있으며 실시간성 보장을 저해하는 원인과 이에 대한 해결 방법들이 제시되었다. 그러나 유닉스와 같은 일반 운영체제와 실시간 운영체제상의 네트워크 디바이스 드라이버에서의 우선 순위 역전 현상으로 인해 네트워크 통신에서의 실시간성 보장을 어렵게 한다[15-16]. 그리고, 일반 운영 체제상의 네트워크에서의 실시간성 보장을 위한 시도가 이루어 졌으며[12-13], RT-Mach상에서의 실시간성 보장을 위한 프로토콜 처리 방법에 대한 연구가 있었다[17-18]. 그리고, 인터럽트 서비스 루틴과 커널 쓰레드에서의 네트워크 패킷 처리로 발생하는 우선 순위 역전 현상을 해소하기 위한 연구가 있었다[19]. 그러나, IEEE 1394 상에서의 실시간성 보장을 위한 연구로는 응용 레벨에서의 실시간성 보장을 위한 간단한 버퍼 제어 기법[22]등이 있을 뿐이며 디바이스 드라이버나 프로토콜 처리 방법에 대한 연구는 아직 발표되지 않았다.

Ⅲ. 요구 분석 및 설계

1. IEEE 1394 실시간 디바이스 드라이버 설계 요건

IEEE 1394의 높은 대역폭과 QoS 지원 특성을 반영하기 위해서는 1394 네트워크 자체의 실시간성 지원과 함께 각 노드의 네트워크 어댑터를 제어하는 디바이스 드라이버와 이를 이용해 프로토콜을 처리하는 소프트웨어에서의 실시간성이 보장되어야 한다. 그러나 기존 형식의 디바이스 드라이버는 1394의 특성을 반영하기에는 다음과 같은 문제점이 있어 새로운 방법의 시도가 필요하다.

(1) 우선 순위 기반의 패킷 처리 구조

현재 사용되고 있는 대부분의 네트워크 디바이스 드라이버[14]는 패킷 처리에 있어 FIFO 큐 방식을 사용하여 패킷이 가지는 우선 순위와 무관하게 패킷이 발생한 순서대로 처리하고 있다. 그러나 실시간 응용에서는 네트워크을 통해 송수신 되는 패킷은 우선 순위를 가지고 있어 패킷을 큐에 들어온 순서대로 처리하게 되면 먼저 큐에 들어온 낮은 우선 순위의 패킷에 의해 그보다 뒤에 들어온 높은 우선 순위의 패킷 처리가 지연 될 수 있다. 이는 높은 우선 순위를 가진 패킷의 처리 시간을 예측할 수 없게 만들며 데드라인을 만족시키지 못하는 상황을 발생시키게 한다. 그리고 패킷의 처리 시간이 처리할 패킷의 우선 순위가 아닌 큐에 대기중인 패킷의 양에 따라 결정되므로 네트워크 부하량이 많을 때와 적을 때의 패킷 처리 시간의 변이가 커지게 된다. 이러한 FIFO 큐를 사용함으로써 발생하는 문제를 해결하기 위해서는 우선순위 큐를 사용하여 우선 순위에 따라서 패킷을 처리하여야 하며 패킷을 처리하는 태스크는 높은 우선 순위 패킷 큐를 처리하는 태스크에 의해 선점될 수 있어야 하고 사용자 태스크와 함께 스케쥴링 되어야 한다.

(2) 등시성 전송에서의 QoS 보장을 위한 버퍼 관리와 스케쥴링 알고리즘

IEEE 1394에서는 대량의 멀티미디어 자료 전송을 위해 QoS를 지원하는 등시성 전송 방식을 지원하고 있다. 응용이나 외부 장치로부터 생성된 등시성 스트림은 송신용 등시성 DMA 버퍼로부터 네트워크 어댑터를 통해 네트워크으로 전송되며, 수신은 등시성 스트림이 네트워크 어댑터를 통해 수신용 등시성 DMA 버퍼에 저장되는 형식을 취한다. 등시성 전송 방식에서 송수신 DMA 버퍼가 DMA에 의해 채워지거나 비워졌을 때는 인터럽트를 통해 네트워크 디바이스 드라이버에 보고되며 이때 디바이스 드라이버는 계속해서 스트림 송수신하기 위해 다중의 DMA 링 버퍼를 사용할 수 있다. IEEE 1394의 등시성 전송 방식은 전송이 이루어 지기 전에 네트워크으로부터 필요한 대역폭과 자원을 할당받게 되며, 실제 송수신은 DMA 버퍼를 통해 이루어지게 된다. 네트워크 디바이스 드라이버는 DMA 버퍼의 크기와 네트워크에 할당된 대역폭을 이용해 인터럽트가 발생하는 주기를 계산할 수 있으며 이 주기는 등시성 스트림을 처리하는 태스크의 주기가 되고 이 주기를 기반으로 한 등시성 스트림 처리 태스크들 간에 우선 순위 스케쥴링이 필요하다.

(3) 인트럽트 지연 시간 단축과 우선 순위 역전 방지 구조

IEEE 1394 버스 상태의 변화와 네트워크 어댑터의 상태 변화는 인터럽트를 통해 디바이스 드라이버에 보고된다. 인터럽트 서비스 루틴(ISR)이 동작 중일 때는 현재 처리 중인 인터럽트 보다 낮은 우선 순위의 인터럽트가 마스크(mask)되며 운영체제에 따라서는 모든 인터럽트가 마스크되기도 한다. 이로 인해 인터럽트 처리 루틴에서의 처리시간 증가는 인터럽트 지연 시간를 증가시키거나 인터럽트를 놓치게 만드는 주요 원인으로 작용한다. 그리고 인터럽트 서비스 루틴과 커널 쓰레드는 현재 진행중인 모든 소프트웨어 태스크 보다 높은 우선 순위로 처리되므로 낮은 우선 순위를 가진 패킷이 수신되어 발생한 인터럽트 서비스 루틴도 높은 우선 순위 패킷을 처리하고 있는 태스크를 선점하므로 긴 인터럽트 서비스 루틴의 처리시간은 인터럽트 지연 시간 증가와 함께 우선 순위 역전 현상도 발생시키는 원인이 된다. 인터럽트가 발생할 때 마다 수신된 패킷의 우선 순위에 상관없이 언제나 인터럽트 서비스 루틴이 동작하게 되므로 인터럽트 서비스 루틴에서의 처리 시간을 최소화하는 것이 필요하며 이를 위해서는 인터럽트 서비스 루틴에서 패킷을 직접 처리하지 않고 인터럽트 서비스 루틴에서 해야할 작업과 그렇지 않은 작업을 나누어 처리하여야 한다. 인터럽트 서비스 루틴에서 처리가 완료되지 않은 패킷은 패킷의 우선 순위에 맞는 우선 순위를 가진 태스크에서 우선 순위에 의해 처리하여 인터럽트와 커널 쓰레드에서의 패킷 처리로 인해 발생하는 우선 순위 역전 현상과 인터럽트 지연 시간의 증가를 줄일 필요가 있다.

2. 제안된 소프트웨어 구조의 개요

이 논문에서 제안하는 네트워크 디바이스 드라이버는 IEEE 1394 네트워크을 통해 다른 노드와 통신하는 응용 프로토콜과 네트워크 어댑터 사이에 위치하게 되며, 1394 네트워크을 통해 다른 노드로부터 수신되는 패킷을 응용 프로토콜로 예측된 시간 안에 전달하고 응용 프로토콜로부터 요청 받은 패킷을 1394 네트워크을 통해 예측된 시간 안에 상대 노드로 전달하는 역할을 한다. 그림 3은 IEEE 1394 프로토콜 스택 전체에서 제안하는 1394 네트워크 디바이스 드라이버의 위치를 나타내는 구성도이다.

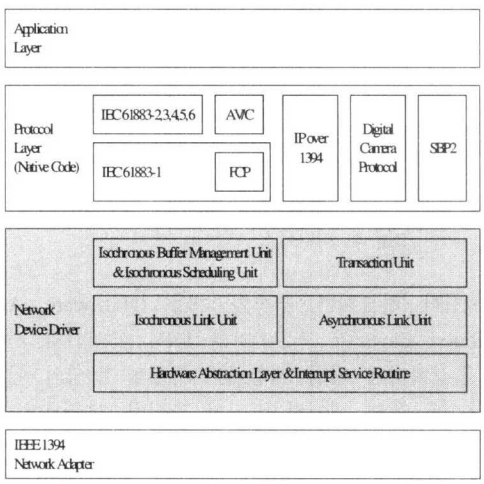


그림 3. IEEE 1394 프로토콜 스택에서의 디바이스 드라이버 계층

여기서 IEEE 1394 네트워크 어댑터는 1394 네트워크 인터페이스의 하드웨어 부분을 나타내고 네트워크 디바이스 드라이버 부분은 제안하는 네트워크 디바이스 드라이버가 위치하게 된다. 그리고 프로토콜 계층은 네트워크 디바이스 드라이버를 통해 통신하는 프로토콜들로 IEC 61883[4]과 AV/C[5]는 디지털 캠코더, DVCR등을 제어하고 멀티미디어 스트림을 전송할 때 이용되는 프로토콜이며 FCP(Function Control Protocol)는 AV/C 프로토콜이 디지털 장비

를 제어할 때 사용되는 하위 프로토콜이다. IP over 1394는 1394 네트워 상에서 IP를 지원하는 프로토콜이며 Digital Camera Protocol[6]은 1394 디지털 카메라를 제어하기 위한 프로토콜이고 SBP2는 프린터와 디스크 드라이버와 같은 디바이스의 제어 명령, 데이터, 상태 정보등을 전송하기 위한 프로토콜이다. 이처럼 디바이스 드라이버 위의 프로토콜 계층에는 다양한 프로토콜이 존재할 수 있다. 응용 계층은 이상에서 지원되는 프로토콜을 이용하여 실제 작업을 수행하는 응용들이 존재한다.

그림 4는 앞에서 설명된 제안하는 IEEE 1394 네트워 디바이스 드라이버의 전체 구조를 나타낸다.

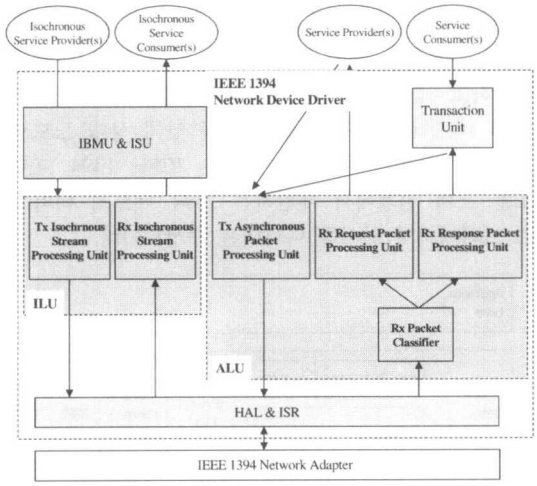


그림 4. 디바이스 드라이버 전체 구조

여기서 하드웨어 추상화 계층 (Hardware Abstraction Layer)은 서로 다른 1394 어댑터의 기능을 추상화하여 상위 계층에서 동일한 방식의 인터페이스를 통해 제어하기 위한 추상화 계층이며 인터럽트 서비스 루틴(Interrupt Service Routine)은 어댑터로부터 발생한 인터럽트를 처리하는 루틴이다. 1394 네트워 어댑터는 하드웨어 추상화 계층을 통해 동일한 방식으로 접근되며 1394 네트워의 변화는 인터럽트 서비스 루틴을 통해 디바이스 드라이버에 전달된다. 링크 계층은 비동기 링크 유닛(Asynchronous Link Unit)과 등시성 링크 유닛(Isochronous Link Unit)으로 나누어 지고, 비동기 패킷이 수신되면 비동기 수신 패킷 분류기(Rx Packet Classifier)를 통해 패킷이 요청 패킷과 응답 패킷으로 분류되어 각각 비동기 수신 요청 패킷 처리기와 비동기 수신 응답 패킷 처리기로 전달된다. 트랜잭션 유닛(Transaction Unit)은 트랜잭션을 발생시키고 트랜잭션 중 발생한 에러를 처리하고 트랜잭션의 결과를 상위 계층으로 전달하는 기능을 담당한다. 그리고 서비스 제공자(Service Provider)는 1394를 통해 다른 노드의 트랜잭션에 응답하거나 등시성 스트림을 제공하는 태스크를 나타내며 서비스 소비자(Service Consumer)는 다른 노드로 트랜잭션을 요청하거나 다른 노드로부터 발생된 등시성 스트림을 받아들이는 태스크를 나타낸다. 서비스 제공자와 소비자는 1394 링크 계층과 트랜잭션 계층 위에서 A/V 스트림 제공이나 기기 제어와 같은 서비스를 제공하기 위한 프로토콜이 구현되는 부분을 나타낸다.

송신 패킷은 패킷을 전송하는 태스크에 의해 패킷의 우선 순위가 결정되고, 다른 노드로부터 수신된 패킷의 경우는 수신 패킷 분류기를 거치면서 수신된 패킷에 우선 순위가 할당된다. 링크 계층과 트랜잭션 계층의 각 유닛은 우선 순위 큐를 포함하고 있어 패킷을 우선 순위에 따라 처리하므로 패킷 처리시 발생하는 우선 순위 역전 현상을 방지하는 구조를 가진다.

이어지는 장에서는 네트워 디바이스 드라이버의 각 부분에 대해 상세 설계 내용에 대하여 설명하도록 하겠다.

3. 비동기 링크 유닛(Asynchronous Link Unit)

1394 패킷이 수신되면 인터럽트를 통해 네트워 디바이스 드라이버에 보고된다. 인터럽트가 발생하게 되면 현재 동작 중인 태스크가 중단되고 디바이스 드라이버에서 제공하는 인터럽트 서비스 루틴이 동작하게 되며 인터럽트 서비스 루틴이 동작 하는 동안은 다른 인터럽트의 발생이 마스크 되므로 긴 인터럽트 서비스 루틴의 처리시간은 인터럽트 지연을 증가시키는 원인이 되며 인터럽트 지연 시간의 증가는 네트워의 상태나 어댑터의 상태변화에 대한 소프트웨어의 응답을 지연시켜 긴급한 이벤트에 대한 데드라인을 만족시키지 못하게 한다. IEEE 1394와 같이 고속으로 동작하는 네트워에서는 인터럽트가 발생하는 주기가 높으며 인터럽트 지연이 시스템에 미치는 영향이 더욱 커지게 된다.

인터럽트 지연 시간을 줄이기 위해서는 인터럽트 서비스 루틴의 수행 시간을 단축하여야 하며 이를 위해서 인터럽트 서비스 루틴에서 직접 이벤트의 처리를 하지 않고 단지 이벤트의 종류를 분석하고 비동기 패킷 수신과 등시성 패킷 수신 그리고 네트워 상태 변화 이벤트 등을 구분하여 해당하는 모듈

에 이벤트를 전달하는 기능만을 담당하도록 한다. 그리고 이벤트의 실제 처리는 상위 태스크에서 이루어 지게 하여 인터럽트 서비스 루틴에서 처리할 일의 양을 줄인다. 비동기 패킷 수신의 경우는 패킷의 종류가 다양하고 포맷이 복잡하며 패킷의 목적지를 찾기 위해 테이블을 참조해야 하는 등 비교적 많은 시간이 걸리는 작업이 있으므로 인터럽트 서비스 루틴에서 패킷을 처리하지 않고 그림 5와 같이 수신 패킷 분류기로 처리를 넘기도록 하여 인터럽트 서비스 루틴에서의 불필요한 시간 지연을 줄이고 있다.

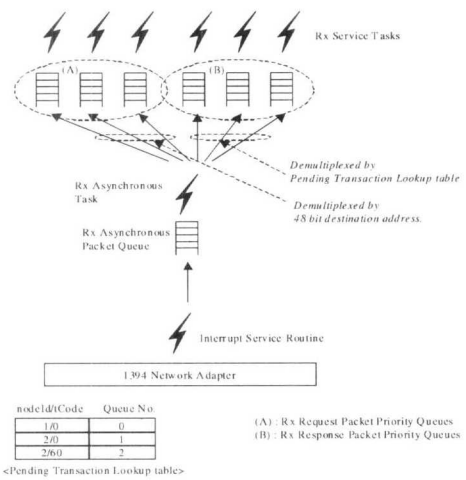


그림 5. 비동기 패킷 수신 유닛

이와 같이 수신된 비동기 패킷은 패킷 분류기로 전달 되며 패킷 분류기는 수신된 패킷의 종류에 따라 패킷 분류기의 수신용 비동기 큐는 수신된 비동기 패킷을 FIFO형식으로 저장하여 패킷의 우선 순위와 상관없이 먼저 수신된 패킷을 먼저 처리하도록 한다. 그 이유는 아직 수신된 비동기 패킷의 우선 순위가 결정되지 않았기 때문이다. 그러나 FIFO 큐로 인해 먼저 들어온 낮은 우선 순위의 패킷에 의해 높은 우선 순위의 처리가 지연된다면 높은 우선 순위의 패킷의 처리 시간을 예측할 수 없게 된다. 이를 방지하기 위해서 패킷 분류기의 Rx Asynchronous Task는 수신된 패킷의 우선 순위를 결정하여 이를 기반으로 비동기 수신 요청 패킷 처리기와 비동기 수신 응답 패킷 처리기의 우선 순위 큐에 넣는다. 수신된 비동기 패킷은 트랜잭션 요청 서브액션용으로 사용되는 요청 패킷과 트랜잭션의 응답 서브액션용으로 사용되는 응답 패킷으로 나눌 수 있으며 패킷의 종류에 따라 수신된 패킷의 우선 순위를 결정하는 방법과 해당하는 우선 순위 큐를 찾는 방법이 달라진다

트랜잭션의 요청 서브액션용 패킷인 경우는 요청 패킷의 목적 오프셋 주소(48bit) 필드를 이용하며, 트랜잭션의 응답 서브액션용 패킷인 경우는 응답 패킷의 노드 아이디(nodeId) 필드와 트랜잭션 코드(tCode) 필드를 키(key)로 이용하여 Pending Transaction Lookup Table로부터 목적지가 되는 우선순위 큐를 찾는다. Pending Transaction Lookup Table은 드라이버가 관리하는 테이블로 다른 노드로 트랜잭션 요청 패킷을 전송한 후 이 테이블에 그 정보가 등록되고 트랜잭션의 응답 패킷이 수신되면 테이블에서 대응되는 트랜잭션 요청 정보를 찾아 수신 패킷의 목적지를 결정하게 된다.

그림 6은 비동기 패킷 송신 유닛을 나타낸다. 동시에 여러 개의 비동기 패킷 송신을 위해 네트워 어댑터의 자원을 비동기 패킷 송신 처리 유닛에 있는 다수의 Tx Asynchronous Task가 공유하므로 네트워 어댑터의 한정된 자원에 대해 바이너리 세마포어를 이용하여 상호 배제 시킨다. 전송을 위한 어댑터 자원이 남을 때 마다 언제나 가장 높은 우선 순위의 패킷이 자원을 점유하게 되므로, 우선 순위가 높은 패킷부터 전송이 이루어지게 된다. 여기서 발생할 수 있는 우선 순위 역전 시간은 어댑터가 패킷 전송 중에 더 높은 우선 순위의 패킷이 큐에 들어와 대기하는 시간, 즉 한정되 하드웨어 자원 점유에 의한 시간이며 현재 전송 중인 패킷 하나가 전송 완료되는 만큼의 시간이다.

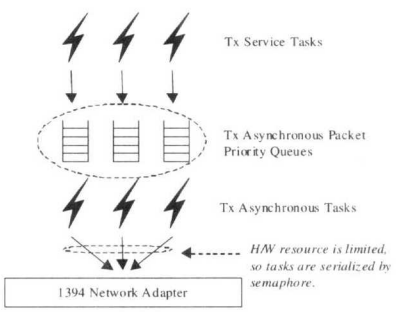


그림 6. 비동기 패킷 송신 유닛

4. 트랜잭션 유닛 (Transaction Unit)

IEEE 1394의 트랜잭션은 요구한 명령이 상대 노드에서 제대로 처리되었는지 확인할 수 있는 기능을 제공하는 통신 모델로 트랜잭션에서의 실시간성을 보장하기 위해 트랜잭션 처리 모듈을 그림 7과 같이 디자인했다.

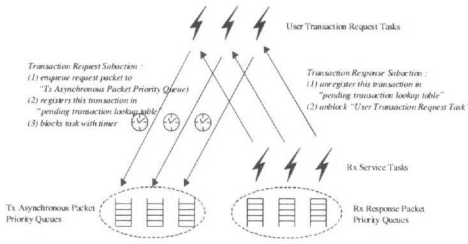


그림 7. 트랜잭션 유닛

트랜잭션을 요구하는 사용자 태스크는 요청 서브액션용 패킷을 패킷의 우선 순위에 따라 적당한 송신용 비동기 패킷 우선 순위 큐에 전달하고 해당 트랜잭션 정보를 PENDING TRANSACTION LOOKUP TABLE에 등록한다. 그리고 트랜잭션을 요구한 사용자 태스크를 블록 시킨다. 이상의 작업은 사용자 태스크의 컨텍스트에서 동작한다. 송신용 비동기 우선 순위 큐에 등록된 패킷은 우선 순위에 따라 Tx Asynchronous Task에 의해 네트웍으로 전송된다.

트랜잭션의 결과는 응답 서브액션용 패킷 형태로 전달된다. 수신된 응답 패킷은 PENDING TRANSACTION LOOKUP TABLE을 참조하여 해당 패킷의 우선 순위를 찾아 적당한 수신용 응답 패킷 우선 순위 큐에 들어오게 되며, Rx Service Task는 이전에 블록 되었던 해당 트랜잭션을 요구한 태스크를 재개시킨다.

5. 등시성 링크 유닛(Isochronous Link Unit)

등시성 통신에서 등시성 패킷은 125 us 마다 수신되며 이 때마다 인터럽트가 발생하게 된다면 CPU 시간의 대부분을 인터럽트 처리에 소비하게 될 것이다. 그러므로 이를 방지하기 위해 등시성 통신을 통해 전달되는 데이터는 네트웍 디바이스 드라이버로 패킷 단위로 전달되지 않고 일정 크기의 DMA 버퍼 단위로 전달된다. 그림 8은 등시성 패킷수신유닛을 나타낸다. 등시성 패킷 수신을 위해서는 수신 전에 버스상의 대역폭 할당과 네트웍 어댑터의 수신 DMA 버퍼 할당과 같은 자원 할당 과정을 거치게 된다. 결정된 대역폭과 수신 DMA 버퍼 크기를 통해 수신 DMA 버퍼가 채워질 때 마다 발생하는 인터럽트의 주기를 계산 할 수 있으며 이를 이용해 등시성 데이터를 처리할 태스크의 우선 순위를 결정할 수 있다. 수신된 등시성 패킷은 비동기 수신 패킷과는 달리 수신된 패킷의 등시성 전송 채널 번호를 이용하여 쉽게 목적지 우선 순위의 큐를 찾을 수 있으므로 인터럽트 서비스 루틴 내에서 직접 수신용 등시성 우선 순위 큐로 전달한다.

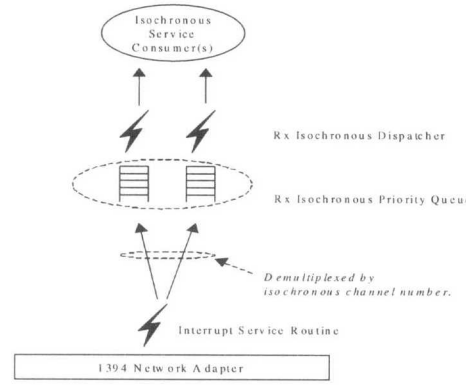


그림 8. 등시성 패킷 수신 유닛

수신용 등시성 디스패처(Rx Isochronous Dispatcher)는 수신된 등시성 스트림을 처리할 함수를 호출하는 기능을 담당하며 이 디스패처의 우선 순위는 다음 장에서 설명될 등시성 스케쥴링 유닛에 의해 결정된다.

그림 9는 등시성 패킷 송신 유닛을 나타낸다. 등시성 전송의 경우는 전송이 이루어지기 전에 미리 채널 할당, 대역폭 결정과 함께 송신 DMA 버퍼와 같은 네트웍 어댑터가 필요로 하는 자원을 미리 할당하므로 일단 성립된 등시성 송신은 필요한 하드웨어 자원을 독점하게 되어 송신용 등시성 패킷 큐에 들어온 패킷을 바로 송신할 수 있다. 그러므로 송신용 등시성 패킷 큐의 개수는 어댑터가 지원하는 DMA 채널의 개수와 동일하게 된다.

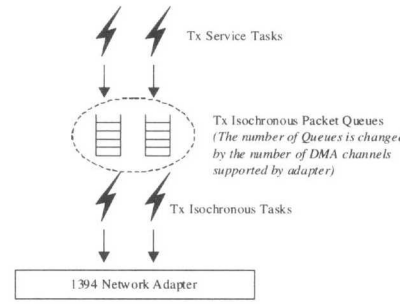


그림 9. 등시성 패킷 송신 유닛

1394의 물리 계층에서는 125 us마다 하나의 등시성 패킷을 전송하지만 네트웍 디바이스 드라이버의 관점에서는 패킷 단위 처리가 아닌 DMA 버퍼 단위로 이루어져 패킷 하나하나를 만들고 전송하는

방식이 아니고 DMA 버퍼에 전송할 데이터를 채우는 작업을 통하여 블록 단위로 전송을 명령하게 된다. 그러므로 송신 DMA버퍼 크기와 대역폭을 알면 송신 DMA 버퍼를 채워야 하는 주기가 결정되며 송신 비동기 태스크(Rx Isochronous Task)는 이 주기를 기준으로 스케쥴링하게 된다. 여기에 대해서는 다음 장의 스케쥴링 유닛에서 설명된다.

6. 등시성 버퍼 관리 및 등시성 스케쥴링 유닛 (IBMU & ISU)

응용은 등시성 전송에 있어 네트웍으로부터 할당받은 대역폭 만큼의 전송 버퍼를 제공해야 하고 수신 시에 수신 버퍼를 제공해야 한다. 전송 버퍼와 수신 버퍼가 소모되는 시간 즉 대역폭에 맞게 버퍼를 계속 제공하기 위해서 응용은 버퍼가 소모되는 주기에 맞게 새로운 버퍼를 제공하여야 한다. 등시성 전송 방식을 통해서는 영상, 음성과 같은 멀티미디어 자료가 주로 전송되므로 전송을 위해 미리 많은 버퍼를 제공하게 되면 높은 전송 지연으로 문제가 된다. 뿐만 아니라 중요한 자원인 메모리를 많이 낭비하게 된다. 이와 마찬가지로 많은 양의 수신 버퍼도 메모리 낭비를 초래하게 된다. 그러므로 가능한 최소의 메모리 버퍼를 준비는 것과 준비된 버퍼를 정확한 시간에 네트웍 어댑터에 전달하는 것이 중요하다.

제안하는 네트웍 디바이스 드라이버에서의 메모리 버퍼는 링 구조를 가지게 구성되어 있으며 하나의 버퍼가 전송되고 있는 동안 다음 버퍼에서는 다음에 전송할 내용을 구성하는 방식을 사용한다. 그리고 전송이나 수신에 사용된 버퍼는 처리가 완료되면 계속 재사용된다. 이때 링 버퍼에 사용될 버퍼의 크기와 개수는 등시성 전송이 이루어지기 전에 가변적으로 구성될 수 있게 구성하여 응용에 따라 그 최적의 링 버퍼 크기를 유지할 수 있도록 했다.

등시성 전송 방식에서는 전송 전에 네트웍 대역폭을 할당하게 되고 네트웍 어댑터는 버스 마스터로 동작하며 DMA를 사용하여 준비된 버퍼를 전송하게 된다. 전송이 완료되면 다음 버퍼를 전송하게 되며 각 전송이 완료될 때 마다 인터럽트를 통해 전송이 완료되었음을 디바이스 드라이버에 보고한다. 그러므로 버퍼의 크기와 대역폭을 이용하면 버퍼를 제공하는 태스크의 주기는 다음의 계산식을 이용해 계산 할 수 있다.

주기(1/sec) = 대역폭(bps) / 버퍼크기(byte)

등시성 송수신이 제대로 이루어지기 위해선 이 주기마다 새로운 버퍼가 준비되어야 한다. 동시에 여러 개의 등시성 전송과 수신이 이루어질 경우 각 태스크는 앞의 방법으로 계산된 주기에 의해 스케쥴링 될 수 있다. 여기서는 주기가 높은 태스크가 높은 우선 순위를 가지는 레이트 모노토닉 스케쥴링 알고리즘을[16] 사용하여 등시성 태스크의 우선 순위를 할당하였다.

7. 하드웨어 추상화 및 인터럽트 처리 유닛 (HAL & ISR)

1394 네트웍 또는 네트웍 어댑터 상태 변화 등 외부의 이벤트는 인터럽트의 형식으로 시스템 소프트웨어인 네트웍 디바이스 드라이버에 보고된다. CPU는 인터럽트가 발생하면 현재 처리하고 있는 사용자 태스크를 중지시키고 인터럽트에 해당하는 인터럽트 서비스 루틴을 수행하게 된다. 즉, 높은 우선 순위의 패킷을 처리하고 있는 중에 낮은 우선 순위의 패킷을 수신한 경우도 인터럽트 발생에 의해 높은 우선 순위 패킷의 처리를 중단하고 낮은 우선 순위 패킷 수신에 대한 인터럽트 처리부터 수행해야 된다. 즉, 인터럽트의 발생은 해당 패킷의 우선 순위를 무시하고 인터럽트 서비스 루틴의 동작을 요구하기 때문에 이로 인한 우선 순위 역전 현상이 발생하게 된다.

그리고, 일반 운영체제의 경우 인터럽트가 발생하게 되면 해당 인터럽트나 낮은 인터럽트 또는 모든 인터럽트를 마스크하여 현재 인터럽트가 처리되고 있는 중에 새로운 인터럽트의 발생이 CPU에 보고되지 않도록 하고 있다. 이런 특성으로 인해 만약 인터럽트 발생시 인터럽트의 처리 시간이 길어 질수록 인터럽트 발생 후 이를 처리할 인터럽트 서비스 루틴이 불릴 때까지의 시간즉, 인터럽트 지연 시간이 증가하게 되어 긴급히 처리되어야 할 인터럽트의 데드라인을 지키지 못하게 만든다.

그러므로, 인터럽트 서비스 루틴에서의 긴 처리 시간으로 인해 발생하는 우선 순위 역전 현상과 인터럽트 지연 시간 증가를 막기 위해서는 패킷 수신 시 처리해야 할 작업을 인터럽트 서비스 루틴 내에서 해야 할 작업과 그렇지 않은 부분으로 세분화 함으로써 인터럽트 레벨에서의 패킷 처리 시간을 줄여야한다.

패킷이 수신되면 해야 할 일을 세분화해 보면 먼저 다음 수신을 위해 패킷 버퍼를 할당 하고 다음으로 수신된 패킷의 유효성 여부 확인하고 주어진 패킷의 헤더를 분석하여 처리할 패킷 핸들러가 패

킷을 처리하도 하고, 사용이 끝난 패킷 버퍼는 재사용을 위해 사용을 해제하는 것이다. 여기서 말한 모든 작업을 인터럽트 안에서 처리하게 된다면 인터럽트 처리 시간이 길어져 앞에서 문제로 제기한 인터럽트에 의한 우선 순위 역전 현상이 발생하는 시간의 증가와 인터럽트 지연 시간의 증가를 발생시킨다. 그래서 패킷 수신 시 인터럽트 서비스 루틴에서는 다음 수신을 위해 패킷 버퍼를 할당하는 작업과 수신된 패킷을 상위 태스크가 사용할 수 있도록 큐에 넣는 작업만을 하고 인터럽트 서비스 루틴이 아닌 상위 태스크에서 패킷 처리에 관련된 나머지 일을 하게 한다.

Ⅳ. 구현 및 성능 평가

1. 구현 환경

본 논문에서 제안한 구조에 따라 Wind River사의 실시간 운영 체제인 VxWorks[23] 상에서 네트워크 디바이스 드라이버를 구현하였다. 구현에 사용된 1394 네트워크 인터페이스 카드는 Adaptec사에서 개발한 AHA-8920과 AHA-8945이며 이 인터페이스 카드는 1394의 링크 계층과 PCI 인터페이스를 하나의 칩으로 구현한 AIC-5800[8]을 사용하고 있다. AIC-5800 칩은 200Mbps까지의 전송속도를 지원하며 6개의 독립적인 DMA 채널을 지원하고 이 중 비동기 전송과 수신을 위해 각각 하나씩, 등시성 전송과 수신을 위해 각각 두개씩 DMA 채널을 할당하고 있으며 비동기 전송, 등시성 전송, 수신을 위해 각각 128 워드 (32비트) 크기의 FIFO를 포함하고 있다.

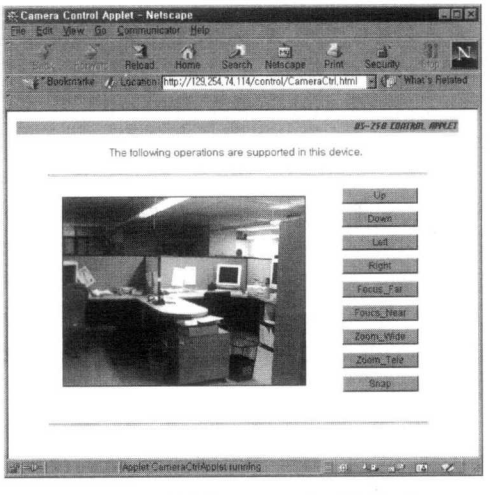


그림 10. 디지털 카메라 프로토콜을 사용한 테스트 프로그램

네트워 어댑터는 물리 계층과 링크 계층까지를 하드웨어로 처리하고 있으며 구현된 네트워 디바이스 드라이버에서는 네트워 어댑터를 제어하는 하드웨어 추상화 및 인터럽트 처리 유닛와 비동기 송수신 유닛, 등시성 송수신 유닛, 트랜잭션 계층을 구현하였다. 그리고 구현된 네트워 디바이스 드라이버 상위에 FCP와 디지털 카메라 프로토콜, AV/C 프로토콜등을 구현하여 IEEE 1394를 지원하는 AV기기를 제어하는 프로그램을 작성하였다. 그림 10은 디지털 카메라 프로토콜을 이용하여 디지털 카메라를 제어하는 프로그램의 화면이다.

2. 성능 평가

본 논문에서 제안한 구조에 따라 제작된 네트워 디바이스 드라이버의 성능 평가를 위해 그림 11과 같이 두 대의 PC를 이용하여 IEEE1394 네트워을 구성하였다. 두 PC 사이는 200Mbps까지 지원하는 1394카드인 AHA-8945를 사용하여 연결하였다.

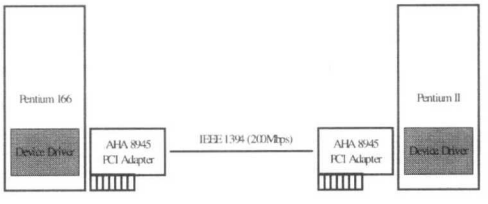


그림 11. 성능 평가를 위한 시스템 구성

다음과 같이 네트워 디바이스 드라이버를 두 가지 형태로 구성하였으며 본 논문에서 제안한 구조를 사용한 네트워 디바이스 드라이버와 기존의 네트워 디바이스 드라이버의 성능을 비교하기 위해 각각에 대하여 라운드 트립 시간 및 변이 실험을 하였다.

▷사례 1: FIFO 큐를 사용하여 패킷을 처리하는 경우

▷사례 2: 논문에서 제안한 우선 순위 큐를 사용하여 패킷을 처리한 경우

실시간 제어가 필요한 응용에서는 다른 노드에 제어 명령 또는 상태 검사 명령을 보내고 그 결과에 따라 다음 동작을 결정해야 하는 경우가 발생하며 이런 경우에 라운드 트립 시간의 예측은 실시간 시스템 구성의 중요한 디자인 요소로 작용한다. 제안한 디바이스 드라이버의 구조가 이와 같은 응용에 적합한 지 판단하기 위해 라운드 트립 시간과

그 변이를 측정하는 실험을 하였다.

이 실험을 위해 그림 12와 같이 두 대의 컴퓨터 PC1과 PC2를 IEEE 1394 네트워크으로 연결하고 PC1에서 PC2로 패킷을 전송하도록 하였다. PC2의 에코 서버는 PC1으로 부터 수신된 패킷에 응답하여 PC1으로 응답 패킷을 되돌리도록 하였다. PC1의 전송 태스크에서 패킷을 전송하기 전 시간을 측정하고 수신 태스크에서 패킷이 수신될 때의 시간을 측정하여 두 시간 차를 라운드 트립 시간으로 하였다.

PC1에는 세 종류의 우선 순위를 가지는 패킷 전송 태스크와 수신 태스크를 각각 세 개씩 동작시키고 PC2에서도 대응되는 세 종류의 우선 순위를 가지는 에코 서버 세 개씩 동작시켰다. PC1의 높은 우선 순위의 패킷 전송 태스크가 전송한 패킷은 PC2의 높은 우선 순위의 에코 서버가 응답하게 하였고 낮은 우선 순위의 패킷은 낮은 우선 순위의 에코 서버가 응답하게 하였다.

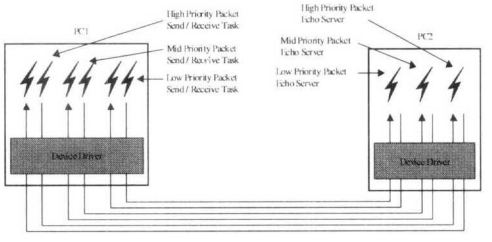


그림 12. 라운드 트립 시간 및 변이 실험

이 실험에서는 PC1의 세 개의 패킷 전송 태스크가 각각 1000개의 패킷을 생성하도록 하여 각 패킷에 대한 라운드 트립 시간을 측정하였으며 사례 1과 사례 2의 결과는 표 1과 같다.

이 실험 결과로 보면 사례1 의 경우 최대 라운드 트립 시간이 1584 us인데 비해 사례2-높은 우선 순위 의 경우는 945us로 제한됨을 알 수 있고,

표 1. 라운드 트립 시간 및 변이 (단위: micro second)

	평균	최대	최소	변이	표준편차
사례1	685.386	1584	601	983	120.4323
사례2 높은우선순위	675.644	945	609	336	47.9328
사례2 중간우선순위	694.509	1267	604	663	127.845
사례3 낮은우선순위	706.072	1588	599	989	164.2514

사례 1의 경우 변이가 983 us임에 비해 사례2-높은 우선 순위 의 경우는 336us로 많이 줄어들었음을 확인할 수 있다. 최대 라운드 트립 시간과 변이가 줄어든 것은 제안한 네트워 디바이스 드라이버가 패킷 처리에 있어 높은 우선 순위의 패킷 처리가 낮은 우선 순위 패킷 처리에 의해 지연되지 않도록 함으로써 얻어진 결과이다.

V. 결 론

본 논문에서는 고속 네트워에 연결된 디바이스에서의 실시간성 보장을 위한 요구 분석과 이를 만족하기 위한 네트워 드라이버의 소프트웨어 구조를 제안하였다.

제안한 네트워 드라이버 구조에서는 우선 순위 기반의 큐를 사용함으로써 송신하거나 수신한 패킷을 처리함에 있어 각 패킷의 특성을 반영한 우선 순위에 따라 높은 우선 순위의 패킷이 낮은 우선 순위 패킷에 의해 처리가 무한히 지연되는 우선 순위 역전 현상을 최소화하였다. 이로 인해 높은 우선 순위의 패킷의 라운드 트립 시간과 변이를 감소시켜 높은 우선 순위 패킷 처리 시간을 예측할 수 있게 되었다. 전송 전에 통신 대역폭을 할당하는 등시성 전송의 경우 전송 전에 계산된 링 버퍼의 크기와 대역폭으로부터 수행 주기를 계산하여 이에 기반한 레이트 모노토닉 스케쥴링을 함으로써 최대한 데드라인을 놓치지 않게 등시성 스트림의 처리 순서를 정할 수 있었다. 그리고, 하드웨어 인터럽트를 통해 네트워 어댑터와 통신하는 디바이스 드라이버에서 실시간성 보장에 영향을 미치는 중요한 요소인 인터럽트 지연 시간을 줄이기 위해 수신된 패킷을 인터럽트 서비스 루틴에서 처리하지 않고 패킷의 우선 순위만을 추출하여 우선 순위 기반의 큐에 넣어 유저 레벨 쓰레드에서 처리하도록 함으로써 인터럽트 콘텍스트에서 처리되는 시간을 줄였다.

제안한 네트워 드라이버 구조의 성능을 평가하기 위해 제안한 구조를 바탕으로 VxWorks 실시간 운영체제 상에서 Adaptec사의 AIC-5800 칩을 이용한 IEEE 1394 네트워 디바이스 드라이버와 실시간 서비스를 요구하는 IEEE 1394 응용을 제작하여 라운드 트립 시간과 변이 실험을 하였으며 그 결과를 통해 제안한 네트워 드라이버 구조가 실시간성을 보장함을 확인할 수 있었다.

IEEE 1394가 현재는 디지털 캠코더나 디지털 VCR, 디지털 TV와 같은 제한된 영역에서 주로 사

용되고 있지만 VESA Home Network, DAVIC, HAVi등 각종 홈 네트워 관련 단체에서 표준 네트워으로 채택하였으며 머지않아 제어 계측과 실시간 CORBA와 같이 실시간성 보장을 요구하는 많은 응용이 생겨날 것으로 기대되어 제안한 실시간성을 보장하는 네트워 드라이버의 구조가 다양한 곳에 적용될 수 있으리라 본다.

참 고 문 헌

[1] Gerard O Dricoll, Essential Guide to Home Networking Technologies, Prentice Hall PTR,2000

[2] Dutta-Roy.A, "Networks for Home", *IEEE Spectrum*, Volume 36, December 1999.

[3] IEEE 1394a, Draft Std. For a High Performance Serial Bus(Supplement), Mar., 1998.

[4] IEC 61883, Consumer audio/video equipment Digital interface, 1998

[5] AV/C Digital Interface Command Set General Specification Version 3.0 April 15, 1998

[6] 1394-based Digital Camera Specification Version 1.20 July 23, 1998

[7] D. Anderson, FireWrie System Architecture: IEEE 1394, Addison-Wesley, 1998

[8] AIC-5800 PCI-to-1394 Controller Chip Data Book and Design-In Handbook, Adaptec

[9] IEEE1394 TA IICP, Draft Standard for Instrument and Industrial Control Protocols, Draft 1.00(RC2),June 17,1999,

[10] IEEE1394 TA IICP488, Draft Specification for IEEE488 Communications using the Instrument and Industrial Control Protocol over IEEE 1394, August,2 1999

[11] LonTalk Protocol Specification Version 3.0, 1994.

[12] Fred Kuhns, Douglas C. Schmidt, and David L. Levine, The Design and Performance of a Real-time I/O Subsystem, *IEEE Real-Time Technology and Applications Symposium (RTAS)*, June 1999, pp. 154-163

[13] Christopher D. Gill, David L. Levine, and Douglas C. Schmidt, The Design and Performance of a Real-Time CORBA Scheduling Service, *Real-Time Systems(The Intl Journal of Time-Critical Computing Systems)* Vol. 20, No.2, March 2001

[14] Linux IEEE-1394 Subsystem, http://www.edu.uni-klu.ac.at/~epirker/ieee1394.html

[15] Sadegh Davari and Lui Sha, Sources of Unbounded Priority Inversions in Real-Time Systems and a Comparative Study of Possible Solutions, *ACM Operating System Review*, Vol. 26, No. 2, April 1992, pp.110-120

[16] C. J. Fidge, Real-Time Schedulability Tests for Preemptive Multitasking, *Real-Time Systems*, 1998, pp 61-93,

[17] Chen Lee, Katsuhiko Yoshida, Cliff Mercer and Raj Rajkumar, Predictable Communication Protocol Processing in Real-Time Mach, *Proceedings of the Ream-time Technology and Applications Symposium*, June 1996

[18] Clifford W. Mercer, Jim Zelenka, and Ragunathan Rajkumar, On Predictable Operating System Protocol Processing, Technical Report CMU-CS-94-165, School of Computer Science, Carnegie Mellon University, May 1994

[19] C. W. Mercer and H. Tokuda, An Evaluation of Priority Consistency in Protocol Architectures, *Proceedings of IEEE the 16th Conference on Local Computer Networks*, 1991

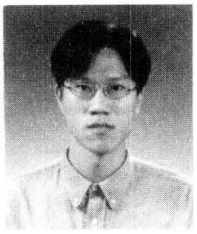
[20] Soon Ju Kang, Jun Ho Park, and Sung Ho Park, ROOM-BRIDGE : Vertically Configureable Network Architecture and Real-Time Middleware for Interoperability between Ubiquitous Consumer Devices in Home, Middleware 2001, *Lecture Notes in Computer Science 2218* (p.232-251)

[21] 오주용, 강순주, IEEE 1394 기반 홈 네트워에서의 코바(CORBA) 기반 미들웨어 설계 및 구현, *한국통신학회 2001년도 추계종합학술발표회*, 2001년 11월,서울

[22] 강성일, 편기현, 이충훈, 이흥규, IEEE 1394 등시성 전송을 위한 선점적 우선순위를 이용한 버퍼 제어 기법 , *제3회 통신 소프트웨어 학술대회 (COMSW98)*, 1998년 7월

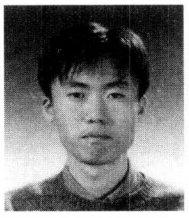
[23] VxWorks Programmers Guide, WindRiver Systems, Mar. 1997

박 동 환(Dong-Hwan Park)



1999년 2월 : 경북대학교 전자공학과 졸업 (공학사)
2001년 2월 : 경북대학교 대학원 전자공학과 졸업(공학석사)
2001년 1월~현재 : 한국전자통신연구원 연구원
<주관심 분야> 홈 네트웍, 내장형 시스템, 분산 미들웨어

임 효 상(Hyo Sang Lim)



1994년 2월 : 경북대학교 전자공학과 졸업 (공학사)
2000년 2월 : 경북대학교 대학원 전자공학과 졸업(공학석사)
1995-1997: 한맥소프트웨어 근무
2000년 1월~현재 : 무이태크 근무
<주관심 분야> IEEE1394, 내장형 시스템, 분산 미들웨어, 자바 프로그래밍

강 순 주(Soon Ju Kang)

1983년 경북대학교 전자공학과 (공학사)
1985년 한국과학기술원 전자계산학과 (공학석사)
1995년 한국과학기술원 전자계산학과 (공학박사)
1985년-1996년 한국원자력연구소 연구원, 핵인공지능연구실 선임 연구원, 전산정보실 실장
2000년 7월 - 2002년 8월, University of Pennsylvania, Dept. of Computer and Information Science, 객원 연구 교수
1996년-현재 경북대학교 전자전기컴퓨터학부 정보통신전공 조교수
<주관심 분야> Real-Time Systems, Software Engineering, Knowledge-Based System
이 메 일 : sjkang@ee.knu.ac.kr