저 사양 시스템에서 오디오 스트리밍을 위한 시스템 설계

신승철[1],정철호[2],한탁돈[3]
연세대학교 컴퓨터과학과[1 2 3]
seungchul.shin@yonsei.ac.kr[1], {brght[2], hantack[3]}@kurene.yonsei.ac.kr

Audio Streaming System Design for Low-Quality Systems

Seung Chul Shin[1], Cheolho Cheong[2], Tack Don Han [3]
Dept. of Computer Science, Yonsei univ. [1 2 3]

요 약

정보통신 기술의 발달로 근래에는 인터넷 방송 서비스가 활성화되었으며 누구든 자유롭게 방송을 제작하거나 청취할 수 있다. 이러한 인터넷 방송을 이용하기 위해서는 PC 를 이용하는 방법이 일반적이지만 오디오 서비스만을 이용할 경우 불편하다. 본 논문에서는 낮은 사양의 시스템에서 인터넷 방송을 청취하기 위한 하드웨어와 소프트웨어 설계 기법 그리고 효율적인 버퍼링 방법을 제안하였다. 제안된 시스템은 8 비트 마이크로 컨트롤러, 32KB 의 메모리, Hardwired TCP/IP Stack 그리고 Hardwired MP3 디코더를 이용하여 설계하였으며 각 작업간의 스케쥴링을 위하여 타이머 인터럽트를 이용하였다. 16KB 의 메모리로 몇 가지 버퍼링 기법을 제안했다. 폴링 방식은 가장 보편적인 방법으로 데이터를 전송받는 작업과 음악을 재생하는 작업이 순차적으로 이루어진다. 이 방법은 데이터 전송과 음악 재생을 동시에 할 수 없기 때문에 타이머 인터럽트를 이용한 버퍼링 모델이 사용된다. 두 번째로 메모리를 두 개의 블록으로 나누어 한 블록에는 데이터를 저장하고 다른 한 블록에는 데이터를 내보내는 '더블 버퍼링'을 제안했다. 세 번째는 메모리 블록을 여러 단계로 나눈 'n-Queue 버퍼링' 기법을 제안했다. 마지막으로 네트워크 상황에 따라서 블록의 개수를 유동적으로 조절하는 '가변 길이 n-Queue 버퍼링' 기법을 제안했다. 이 방법은 네트워크 상황에 따라 메모리의 크기를 유동적으로 할당하기 때문에 메모리 사용률이 높아지는 장점이 있다. 본 논문에서 제안하는 시스템은 운영체제를 사용하지 않았기 때문에 TV 나 오디오 등 다른 시스템에 이식이 용이하므로 다양한 기기에 적용이 가능하다.

Keyword : Buffering, Streaming, MP3, n-Queue, 가변길이 n-Queue

1. 서론

특히 정보통신 기술이 발달하면서 인터넷을 매개로하는 멀티미디어 방송이 늘어나면서 스트리밍 서비스에 대한 많은 연구가 이루어지고 있다. 스트리밍이란 음성이나 영상과 같은 멀티미디어 데이터를 실시간으로 재생하는 기법이다. 스트리밍 서비스는 서버로부터 데이터를 전송 받아서 재생하므로 디스크 용량의 제약을 받지 않는다. 그런데 서버에서 발생하는 비동기적인 성질과 클라이언트에서 발생하는 주기적인 성질의 차이를 줄이기 위하여 멀티미디어 데이터 블록을 미리 읽어들이고(pre-fetch), 잠시 보관하기 위하여 버퍼링을 할 필요가 있다[1]. 현재 인터넷 방송을 청취하기 위해서는 PC 를 이용하는 방법이 일반적이다. 하지만 음악 청취만을 목적으로 하기에는 PC 는 휴대성이나 사용 목적상 적합하지 않고, 현재 시중에 출시된 MP3 플레이어는 인터넷 기반의 인터넷 방송을 청취할 수 없다.

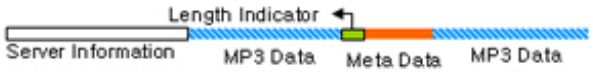
본 논문에서는 기존의 MP3 플레이어와 동일한 시스템에서 인터넷 음악 방송을 청취하기 위한

하드웨어 설계기법과 소프트웨어 개발방법 그리고 버퍼링 모델에 대해서 제안한다. 제안한 시스템은 인터넷을 사용하기 위하여 온칩화된 TCP 스택과 8 비트 마이크로 컨트롤러, 음악을 청취하기 위한 전용 MP3 디코더를 사용한다. 이 시스템은 소프트웨어적으로 처리하던 작업들을 하드웨어로 처리하면서 낮은 사양에서도 인터넷 방송을 청취할 수 있다. 또한 이 시스템을 구현하기 위한 버퍼링 모델들을 설계하였다. 가장 일반적인 폴링 방식에서 타이머 인터럽트를 이용한 방식, 메모리 블록을 두 개로 나뉘어 버퍼링하는 더블 버퍼링 방식과 블록을 여러 개로 나뉘는 n-Queue 버퍼링 시스템, 그리고 네트워크의 상황에 따라 블록의 개수를 유동적으로 조절하는 가변길이 n-Queue 버퍼링 기법을 제안하였다. 각각의 버퍼링 모델은 시스템의 특성과 목적에 맞게 다양하게 적용될 수 있다.

본 논문의 2 절에서는 인터넷 방송 청취를 위해 필요한 기술을 소개하고, 3 절에서는 하드웨어와 소프트웨어 관점에서의 구현 기법과 버퍼링 모델에 대해서 살펴본다. 4 절에서 실험 결과를 제시하고 5 절에서 결론과 추후 과제에 대하여 기술하였다.

2. MP3 기반 인터넷 방송

최근의 인터넷 음악 방송은 TCP 방식의 HTTP 기반의 프로토콜을 이용한 MP3 방송이 일반적이다. MP3 는 순차적으로 압축이 되어있기 때문에 실시간 스트리밍이 가능하며, TCP 기반의 네트워크는 패킷의 손실이 발생하지 않으므로 끊김이 없는 인터넷 방송 수신이 가능하다. MP3 기반의 인터넷 방송은 Null 소프트사의 Shoutcast 방송이 널리 쓰이고 있다. 현재 Shoutcast 홈페이지에 등록된 공식 방송만 11,500 여 개나 되며 개인이 직접 운영하는 비공식 방송까지 합치면 그 수는 훨씬 더 많아진다[2]. Shoutcast 음악 방송은 ICY 프로토콜을 사용하는데 ICY 프로토콜은 HTTP 와 유사하게 GET 명령어를 이용하여 스트리밍 데이터를 전송받는다. ICY 프로토콜은 다음과 같은 형식으로 구성되어있다[3].



[그림 1] ICY 프로토콜의 구조([3]을 수정함)

위의 [그림 1]과 같이 ICY 프로토콜은 먼저 서버의 정보가 위치하고 그 뒤에 MP3 데이터와 길이 표시자 메타 데이터가 순차적으로 위치한다. MP3 데이터의 사이에 위치하는 1 바이트 크기의 길이 표시자와 메타 데이터는 현재 재생중의 방송의 정보를 실시간으로 알리기 위해 사용되며 길이 표시자 * 16Byte 만큼 메타데이터가 표시된다. 이 ICY 프로토콜의 명령어 형식은 다음과 같다.

```
GET / HTTP/1.0\r\n
Host: cast1.mukulcast.com\r\n
User-Agent: winampMPEG/2.9\r\n
Accept: */*\r\n
Icy-MetaData:1\r\n
icy-metaint:8192\r\n\r\n
```

위의 명령어는 HTTP 의 GET 명령어를 사용하며, [그림 2]와 같이 방송 서버의 정보를 가장 먼저 수신한다.

```
ICY 200 OK
icy-notice1:<BR>This stream requires <a href="ht
tp://www.winamp.com/">Winamp</a><BR>
icy-notice2:SHOUTcast Distributed Network Audio
Server/Linux v1.9.2<BR>
icy-name:Republic of Korea Top Radio. MUKULCAST.
COM * KPOP
icy-genre:Republic of KoreaPOP
icy-url:http://www.mukulcast.com
Content-Type:audio/mpeg
icy-pub:1
icy-metaint:8192
icy-br:128
[bluemary@s1 bluemary]$
```

[그림 2] ICY 서버의 정보

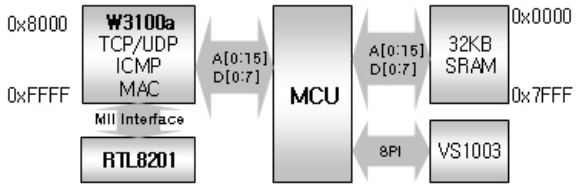
ICY 서버의 정보는 [그림 2]와 같은 형식으로 표시되는데 가장 먼저 icy-notice1, icy-notice2 에서 현재 방송중인 서비스의 종류와 권고사항이 나오며 Content-Type 에서는 스트리밍 데이터의 종류가 표시되고 스트리밍 데이터의 한 프레임이 길이와 초당 전송률이 표시된다. 이 정보를 수신한 후에는 방송 데이터와 길이 표시자 그리고 메타 데이터를 연속해서 전송받는다. 따라서 클라이언트에

서는 스트리밍 데이터와 메타 데이터를 구분해야 한다. [그림 2]의 icy-metaint 필드를 통해서 스트리밍 데이터의 한 프레임의 길이를 계산할 수 있으며 이후 스트리밍 데이터를 수신한다.

3. 인터넷 방송 수신을 위한 시스템 구현

3-1 하드웨어 설계

본 논문에서 제안한 MP3 방송 수신을 지원하는 시스템 구조는 [그림 3]과 같다. 메인 컨트롤러는 아트멜사의 ATmega128 을 사용하였고 11.0592MHz 로 동작을 한다. 외부메모리로 8 비트 데이터 버스와 15 비트 어드레스 버스를 가진 32KB SRAM 을 사용하였고, 네트워크 연결을 위하여 위즈넷사의 W3100a 를 사용하였다. W3100a 는 OSI 계층의 전송계층(Transport Layer)까지 칩으로 구현되어있어서 빠르게 제어를 할 수가 있으며 메모리맵을 통한 다이렉트 방식, 세 개의 데이터 버스를 통한 인다이렉트 방식 그리고 I2C 인터페이스를 지원하기 때문에 다양한 컨트롤러에서 사용할 수 있는 장점이 있다. 구현된 시스템은 메모리맵을 통한 다이렉트 방식을 이용했다.

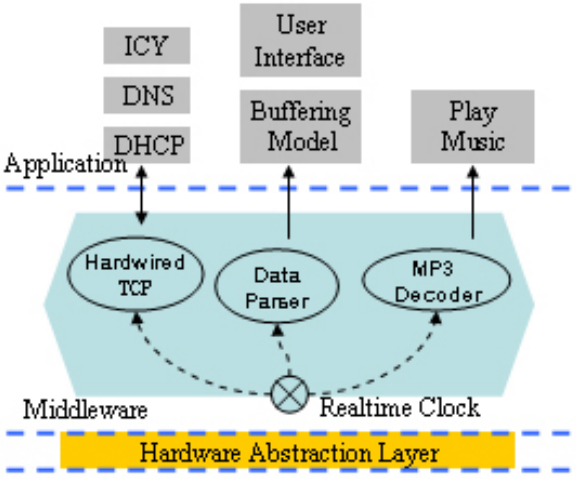


[그림 3] 제안된 시스템의 하드웨어 아키텍쳐

시스템은 [그림 3]과 같이 컨트롤러를 중심으로 SRAM 과 하드웨어 TCP 가 메모리맵 방식으로 연결되어있다. 네트워크에서 전송받은 스트리밍 데이터는 메모리 W3100a 에 위치한 메모리 저장되며 이 데이터들은 버퍼링을 거친 후 재생된다.

3-2 소프트웨어 계층구조

본 논문에서 제안하는 시스템의 소프트웨어 계층 구조는 [그림 4] 와 같이 크게 Hardwired TCP 와 MP3 디코더를 제어하고 전송받은 데이터를 파싱하는 미들웨어 부분과, DHCP, DNS, ICY 등의 소켓, 버퍼링, 음악 재생 등의 어플리케이션의 두 단계로 나뉘어져 있다. 그리고 각 단계의 태스크를 조율하기 위하여 타이머 인터럽트를 이용한 리얼타임 스케쥴러를 구현했다. 본 시스템은 운영체제를 이식하지 않았으며 모든 프로그램은 펌웨어 기반으로 개발되었다.



[그림 4] 소프트웨어 계층 구조

3-3 버퍼링 시스템 설계

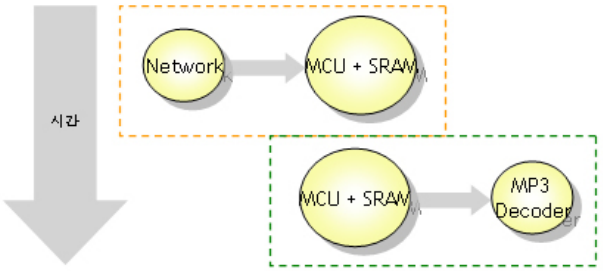
저 사양 시스템에 적용할 수 있는 버퍼링 모델들을 구현했는데, 크게 타이머 인터럽트를 사용했는지 여부와 버퍼에 사용된 메모리 블록의 개수에 따라 폴링 방식 버퍼링 모델, 타이머 인터럽트 방식 버퍼링 모델, 더블 버퍼링 모델, n-Queue 버퍼링 모델로 나뉘며, 메모리의 사용량의 관점에서 가변길이 n-Queue 버퍼링 모델이 있다. 이중 폴링 방식을 제외한 네 가지 모델을 제안하였다.

3-3-1 폴링 방식의 버퍼링 모델

가장 기본적인 버퍼링 모델은 서버에서 데이터를 받아오는 작업과 받아온 데이터를 재생하는 작업이 연속적으로 이루어지는 방식이다. 이런 방식은 루프문을 이용하여 무한 반복을 하기 때문에 폴링 방식이라 불린다. 이 방식은 가장 간단하고 구현이 쉽지만 데이터를 받아오는 작업과 음악을 재생하는 작업이 동시에 수행될 수 없기 때문에 시스템 속도가 느리거나 데이터가 데이터의 전송 대역이 큰 경우 끊김 현상이 자주 발생한다.

[그림 5]는 폴링 방식의 버퍼링 모델을 표현

한 것이다. 세로축은 시간을 나타내고 가로축은 태스크를 나타낸다.

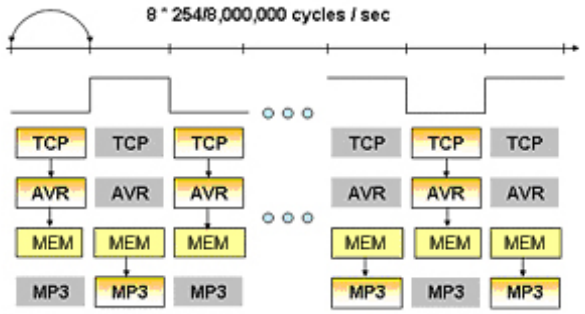


[그림 5] 폴링 방식 버퍼링 모델

컨트롤러를 중심으로 데이터를 받아오는 작업과 음악을 재생하는 작업이 동시에 이루어지지 않기 때문에 저 사양 시스템에서 방송 수신이 목적인 시스템에 적합하지 않다.

3-3-2 타이머 인터럽트를 이용한 버퍼링 모델

폴링 방식의 버퍼링 모델의 단점은 운영체제에서 쓰레드를 이용하면 해결할 수 있다. 그러나 저 사양 시스템에서는 운영체제를 이식하는 것 자체가 오버헤드이다. 따라서 운영체제의 도움 없이 타이머 인터럽트를 이용하는 버퍼링 모델을 제안했다.



[그림 6] 타이머 인터럽트 버퍼링 모델

[그림 6]은 타이머 인터럽트를 이용한 버퍼링 모델을 나타낸 것이다. 가로축은 시간을 나타내며 세로축은 시간에 따른 태스크를 나타낸다. 폴링 방식이 네트워크에서 데이터를 받거나 MP3 디코더에 데이터를 내보내는 작업이 동시에 수행되지 못했던 것이 비하여 이 방식은 타이머 인터럽트가 발생하는 주기마다 두 작업을 동시에 수행하기 때문에 쓰레드를 사용한 것과 유사한 효과가 있다. 타이머 인터럽트를 사용하면 운영체제를 이식하지 않더라도 실시간으로 작업을 수행할 수 있기 때문에 저 사양 시스템에서 방송을 청취하기 위한 방법으로 적합하다.

이 버퍼링 모델의 핵심은 작업들간의 최소 수행 시간을 보장해주는 것에 있다. 작업들간의 수행 시간을 너무 짧게 잡게 되면 본 작업 보다 작업들간의 전환에 많은 CPU 연산을 소모하며, 너무 길게 잡으면 폴링 방식처럼 음악의 끊김 현상이 발생한다. 타이머 인터럽트의 주기는 재생하는 미디어의 초당 전송률(Bitrate)에 의존한다.

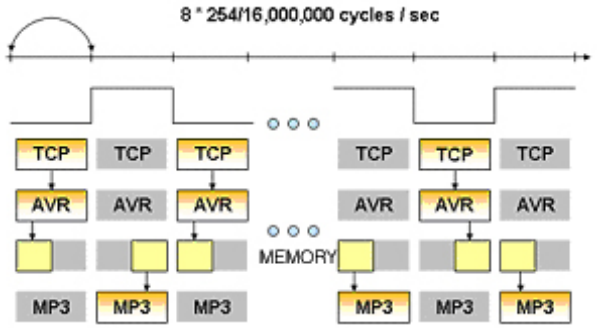
3-3-3 더블 버퍼링 모델

폴링 방식과 타이머 인터럽트를 이용한 버퍼링 모델은 한개의 메모리 큐를 이용한다. 그런데 만약 시스템 메모리가 작을 경우 큰 메모리 블록을 할당하는 것이 불가능 할 경우가 있다. 따라서 메모리 블록을 두 개를 이용하여 데이터 적재와 소비가 번갈아가며 발생하는 더블 버퍼링 모델을 제안했다. 이 기법은 두 개의 큐를 이용하는데 한 개의 큐에는 데이터를 쌓는 작업(Write)이 이루어지고 다른 하나의 큐에는 MP3 디코더로 데이터를 내보내는(Read) 작업이 이루어진다.

이 방식은 그래픽 카드에서 모니터로 데이터를 출력할 때 사용하는 기법을 응용한 것이다. 그래픽 카드에서 실시간으로 모니터에 출력을 하기 위해서는 두 개의 큐 A, B 를 이용한다. 큐 A 에서 출력을 하는 동안 큐 B 에서는 다음 화면 데이터를 읽는다. 그리고 큐 A 에서 출력을 마치면 연속해서 큐 B 의 데이터를 화면에 출력하고 큐 A 는 그 다음 화면 데이터를 읽는다[4]. 이 기법의 핵심은 한쪽 큐를 비울 때의 시간과 다른 큐에서 데이터를 읽어오는 시간이 서로 일치해야 한다는 점이다.

[그림 7]은 더블 버퍼링 모델을 나타낸 것이다. 그림을 보면 메모리 블록이 두 개의 큐로 나뉘어 졌으며 각각 Write 작업과 Read 작업이 번갈아 발생됨을 알 수 있다. 이 기법은 한쪽 큐가 음악 재생을 위해 사용되는 동안에 다른 큐에서는 다음 데이터를 저장하고 있기 때문에 한개의 큐만을 사용하는 버퍼링에 비해서 데이터의 연속성을

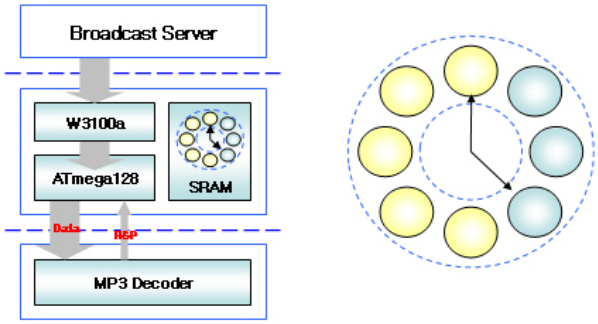
유지하기 쉬운반면 한쪽 큐가 소비되는 시간과 다른 큐에 데이터를 저장하는 시간의 일치성을 유지시키기가 어렵다는 단점이 있다.



[그림 7] 더블 버퍼링 모델

3-3-4 n-Queue 버퍼링 모델

더블 버퍼링 기법의 문제점인 두 개의 메모리블럭간의 시간 동기화 문제는 메모리를 여러 단계로 나누면서 해결할 수 있다. 따라서 메모리 블록을 여러 단계로 나뉘는 n-Queue 버퍼링 모델을 제안하였다.



[그림 8] n-Queue 버퍼링 모델

[그림 8]은 n-Queue 버퍼링 모델은 나타낸 것이다. n-Queue 버퍼링 모델은 큐를 여러 단계로 나누어서 각 큐가 마치 생산자-소비자 관계처럼 스트리밍 데이터가 저장되고(Write) 소비되는 것을 알 수 있다. 현재 Read 중인 큐에서 재생이 완료되면 연속으로 다음 큐에서 권한을 가지고 있는 큐가 전부 소비되면 다음 큐의 데이터를 MP3 디코더로 내보낸다. 더블 버퍼링 처럼 Read 큐와 Write 큐의 시간 동기화를 맞출 필요 없이 Read 를 큐는 Read 가 끝나면 바로 다음 큐의 데이터를 읽어오면 되고 Write 큐는 버퍼가 모두 Full 되지 않는 한 계속해서 데이터를 Write 하기만 하면 된다. 이 기법은 더블 버퍼링 모델 단점이었던 큐들간의 시간 동기화를 해결했다는 장점이 있으며 구조도 복잡하지 않아서 저 사양 시스템에 가장 적합하다.

3-3-6 가변길이 n-Queue 버퍼링

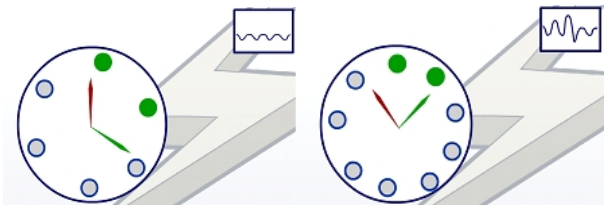
n-Queue 버퍼링은 버퍼의 개수가 고정이었던 것에 비하여 가변길이 n-Queue 버퍼링은 네트워크의 상태에 따라서 버퍼의 개수를 유동적으로 변화시킨다.

가변길이 n-Queue 버퍼링은 네트워크 상황이 좋아서 일정량의 데이터를 끊임없이 전송받을 경우 적은 양의 메모리로도 버퍼링이 가능하다는 가정에 기반한 버퍼링 모델이다. 만역 네트워크 상태가 불량하여 전송받는 데이터의 양이 불규칙할 경우 버퍼의 크기를 늘여서 조금이라도 많은 양의 데이터를 저장해야 할 것이다. 하지만 네트워크 상황이 좋다면 굳이 많은 양의 메모리를 버퍼링에 할당할 필요가 없다. 이 경우 버퍼링에 사용된 메모리를 시스템에 반환한다면 이 메모리는 다른 프로그램에 사용될 수 있으며 시스템 전체적으로 볼 때 메모리 사용이 효율적이다.

[그림 9]는 가변길이 n-Queue 버퍼링을 나타낸 것이다. [그림 9]의 왼쪽은 네트워크 상태가 좋아서 일정량의 데이터를 지속적으로 전송받는 경우를 나타낸 것이며, 오른쪽은 네트워크 상태가

	폴링방식	타이머 인터럽트	더블 버퍼링	n-Queue 버퍼링	가변길이 n-Queue 버퍼링
실시간 여부	X	○			
메모리 양	16KB				
큐의 개수	1 개	1 개	2 개	16 개	4~16 개
장점	구현이 간단하다	가장 쉽게 적용할 수 있다.	연속적인 버퍼링을 가장 빠르게 구현.	큐간의 동기화를 맞춰줄 필요가 없다.	시스템 전체의 관점에서 메모리 효율 상승.
단점	실시간 버퍼링을 지원하지 않음.	시스템에 따라 메모리 할당이 불가능	큐간의 시간동기화에 따른 오버헤드 발생	구현이 복잡	메모리 블록 조정에 오버헤드 소요.

불량하여 전송 받는 스트리밍 데이터의 양이 불규칙한 경우이다.



[그림 9] 가변길이 n-Queue 버퍼링 모델

[그림 9]의 오른쪽 그림 처럼 네트워크의 상황이 좋지 않다면 버퍼의 양을 늘리지만, 왼쪽 그림처럼 네트워크 상황이 좋아서 일정양의 데이터를 지속적으로 받아들일 수 있다면 버퍼의 개수를 줄인다. 반환된 메모리는 다른 프로그램에서 사용될 수 있다.

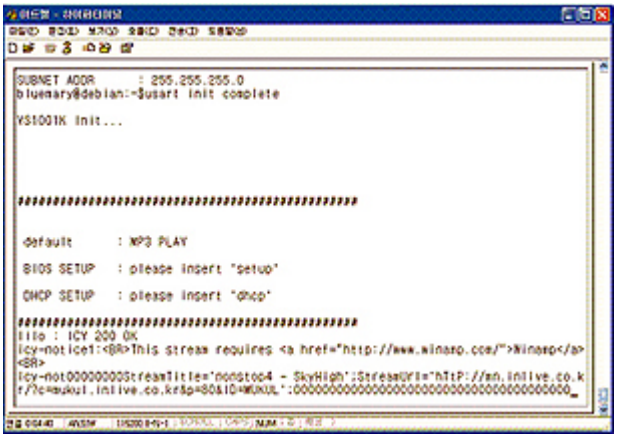
이 버퍼링 모델은 영상 스트리밍과 같이 메모리를 많이 사용해야 하는 경우 이 버퍼링 모델은 유용하게 사용될 수 있을것이다. 하지만 한 시스템에서 여러 프로그램이 수행되는 경우가 아니라면 메모리를 할당하고 반환하는 작업 자체가 오버헤드로 작용될 수 있다. 이 버퍼링 모델은 PDA 같은 멀티 태스킹이 지원되며 메모리의 양이 제한된 시스템에 적합하다.

4. 실험결과

제안된 각 버퍼링 모델의 성능을 측정한다. 성능을 측정하기 위하여 [그림 10]과 같이 논문에서 제안하는 하드웨어를 개발했다.

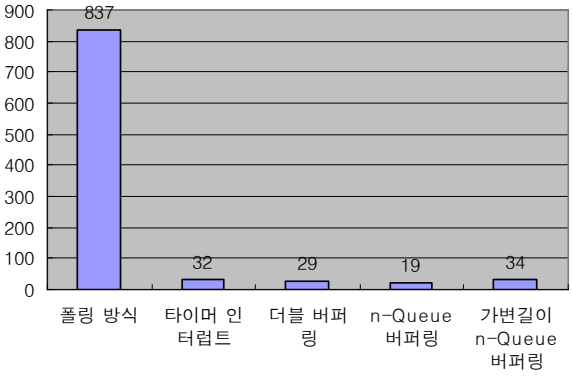
[그림 10] 버퍼링 모델이 탑재된 시스템

개발된 하드웨어는 아트멜사의 8 비트 마이크로 컨트롤러인 ATmega128 을 사용하였으며 32KB 의 SRAM 과 위즈넷사의 Hardwired TCP 스택인 W3100a 를 연결하였고 MP3 디코더로는 VLSI 사의 VS1001K 를 사용하였다. 하드웨어의 상태를 확인하고 조작을 위하여 키패드와 LCD 를 부착했다.



[그림 11] 방송 끊김 측정 프로그램

개발된 하드웨어에서 제안된 버퍼링 모델을 탑재했으며 현재 서비스중인 방송 서버에 접속하여 방송을 청취했다. 방송을 수신하면서 제 때 재생을 하지 못할 경우 PC 와 연결된 시리얼 케이블을 통하여 PC 로 시그널을 전송했으며 [그림 11]과 같은 PC 소프트웨어에서 방송이 끊긴 횟수를 측정하였다. 실험은 각 버퍼링 모델마다 한 시간씩 총 5 회에 걸쳐 진행되었으며 최대치와 최소치를 제외한 나머지 값들의 평균을 측정했다.



[그림 12] 실험 결과

[그림 12]에서 가로축은 제안하는 버퍼링 모델들의 종류를 나타내며 세로축은 각 버퍼링 모델이 한 시간 동안 재생했을 때 끊김이 발생한 횟수를 나타낸다. 위의 결과에서 타이머 인터럽트를 적용한 버퍼링 모델인 타이머 인터럽트 모델, 더

블 버퍼링, n-Queue 버퍼링, 가변길이 n-Queue 버퍼링 모델은 비슷한 성능을 보였다. 실험을 통해서 낮은 사양의 시스템에서는 타이머 인터럽트를 사용하는 것이 인터넷 방송 청취에 유리함을 알 수 있다.

폴링 방식을 제외한 나머지 버퍼링 방식들은 모두 같은 양의 메모리를 사용함에도 불구하고 차이를 보였는데, 측정 당시의 네트워크의 품질 차이도 있겠지만 버퍼링을 위해 메모리 블록 수에 따른 메모리 연산의 차이 때문이라 추정된다

5. 결론 및 추후과제

4 절의 실험결과를 통해서 32KB 의 메모리만을 사용하며 11MHz 로 작동하는 8 비트 시스템에서도 충분히 스트리밍이 가능함을 확인했다. 실험에 적용된 시스템은 운영체제를 사용하지 않았으며 극히 적은양의 메모리로 구현이 되었기 때문에 일반 오디오 시스템이나 가전기기에 적용하는데 큰 무리가 없을것이라 기대된다.

각 버퍼링 모델은 성능이 비슷하기 때문에 시스템의 특성에 맞춰서 적용하면 된다. 타이머 인터럽트 버퍼링 모델은 개발이 쉽지만 시스템에 따라서 메모리 할당이 불가능할 수 있으며 더블 버퍼링 모델은 타이머 인터럽트 버퍼링 모델의 단점을 보완했지만 두 개의 큐간에 발생하는 시간동기화를 맞추는 것이 쉽지 않다.

만약 인터넷 방송만을 목적으로 시스템을 개발한다면 n-Queue 버퍼링 모델을 적용하는 것이 유리하며, PDA 와같이 인터넷 방송 수신 이외에 다른 작업을 동시에 수행하는 경우라면 가변길이 n-Queue 버퍼링 모델이 적당하다. 하지만 끊김 없는 스트리밍 방송을 위해서는 네트워크가 방송 서비스를 위한 최소한의 대역폭이 전제가 되어야한다. 또한 대부분의 버퍼링 모델은 대부분이 메모리 연산으로 이루어져 있는데, 낮은 사양의 시스템일수록 메모리 연산은 오버헤드로 작용한다. 따라서 더 낮은 사양의 시스템에서도 동작할 수 있는 메모리 접근 연구가 필요하다.

감사의 글

본 연구는 교육인적 자원부의 BK21 사업의 지원을 받아 수행되었습니다.

6. 참고문헌

[1] 전용희, "VOD 시스템을 위한 버퍼링 기법," *한국통신학회지* 제 14 권 제 7 호, pp.156-166, 1996.

[2] Shoutcast Corporation, "http://www.shoutcast.com/", January, 2006.

[3] smackFU, "*http://www.smackfu.com/*", January, 2006.

[4] Garcia-Martinez, A, "Minimizing buffer requirements in video-on-demand servers," *EUROMICRO 97. 'New Frontiers of Information Technology'. Short Contributions., Proceedings of the 23rd Euromicro Conference*, pp.68 – 73, 1997.

[5] 권장우, "개선된 MP3 오디오 전송을 위한 SPH/RPH 모듈 구현," *한국멀티미디어학회 춘계학술발표논문집*, pp.338-341, 2003.

[6] 신승철, "Shoutcast Player Based on AVR," 삼성전자 소프트웨어 멤버십, 2004.

[7] Information Technology – Generic Coding of Moving Pictures and Associated Audio : Audio, ISO/IEC 13818-3, "*http://trap.mtview.c a.us /~tom/tech/file-formats/mpeg-2.html*", 1994.

[8] D.James Gemmel, "Multimedia Storage Servers: A Tutorial", *IEEE computer*, 1995.

[9] 손주영, "인터넷에서 고품질 오디오 스트리밍 서비스를 위한 복합적 QoS 보장 기법", *Journal of Korea Multimedia Society* Vol.*7* No.*1*, January 2004, pp.54-63.