SoC 플랫폼 상에서 임베디드 블루투스 오디오 스트리밍 솔루션 개발

김 태 현†

요 약

본 논문에서는 블루투스 무선 링크를 이용한 실시간 오디오 스트리밍을 위해 DSP를 내장한 SoC (System-on-Chip) 플랫폼 상에서 임베디드 블루투스 솔루션의 개발과 최적화에 대해 설명한다. 개발된 솔루션은 이식성을 고려해서 가상 운영체제 상에서 구현된 임베디드 블루투스 프로토콜 스택, 프로파일과 타겟 멀티미디어 SoC의 특성을 이용한 최적화 기법들을 포함한다. 주요 최적화 기법으로는 SoC 내의 스크레치 패드 메모리의 활용을 통한 메모리 접근 최소화, DSP 연산과 병렬 메모리 접근 명령을 이용한 코덱 구현, 무선 통신 환경을 고려한 동적 오디오 품질 조정 등이 있다. 실험 결과는 본 연구에서 제안한 최적화 기법을 적용한 임베디드 솔루션은 별도의 외부 메모리 없이 고품질 오디오 스트리밍을 지원할 수 있음을 보여준다.

키워드 : 블루투스, SoC, 오디오 스트리밍

Development of an Embedded Bluetooth Audio Streaming Solution on SoC Platform

Taehyoun Kim†

ABSTRACT

In this paper, we describe the development and optimization of an embedded Bluetooth solution on an SoC platform for real-time audio streaming over a Bluetooth wireless link. The solution includes embedded Bluetooth protocol stack and profile simplemented on a virtual operating system for portability, and other optimization techniques to fully exploit the benefits of multimedia-oriented SoC. The optimization techniques implemented in this paper are memory access minimization by using on-chip scratch pad memory, codec library optimization with DSP and parallel memory access instruction set, and dynamic audio quality adjustment regarding current wireless link status. Experimental results show that the optimized solution presented in this paper can support high-quality audio streaming without the support of external memory.

Key Words : Bluetooth, System-on-Chip, Audio Streaming

1. 서 론

블루투스는 2.4 GHz 주파수 대역을 이용하여 10 미터 내외의 단거리에서 다양한 기기간 통신을 목적으로 사용되는 저전력, 저비용 무선 통신 기술이다. 에릭슨 (Ericsson) 사에 의해 휴대 전화기용 주변 기기를 연결하는 케이블을 무선 링크로 대체할 목적으로 개발된 블루투스 기술은 음성, 데이터 통신을 동시에 지원하며 다양한 응용 프로그램을 지원할 수 있는 응용 프로파일들이 잘 정의되어 있어 휴대 전화기의 핸즈프리, 무선 전화기, 마우스와 키보드 등 개인용 컴퓨터의 입출력 장치, 개인 휴대 정보 단말기, 유선 LAN을 이용한 인터넷 접속을 위한 액세스 포인트 등 다양한 응용에 활용되고 있는 추세이다[1]. 최근에는 멀티미디어 데이터를 지원하는 다양한 개인 휴대 단말 기기와 가정내 정보 가전 기기의 도입으로 개인 영역에서 블루투스 기술을 이용한 무선 오디오 스트리밍에 대한 관심이 증대되고 있는 추세이다.

이러한 추세에 맞춰 블루투스 기술을 이용한 오디오 스트리밍에 관한 연구가 활발하게 이루어져 왔는데, 대부분의 기존 연구들은 기존 인터넷 망이나 데스크탑과 같은 범용 시스템 환경을 기반으로 오디오 스트리밍을 지원하는 데 초점을 맞추고 있다[2-4]. 본 논문에서는 멀티미디어 응용을

† 종신회원 : 서울시립대학교 기계정보공학과 조교수
논문접수 : 2006년 6월 28일, 심사완료 : 2006년 10월 17일

위해 특화된 SoC 상에서 블루투스 오디오 스트리밍을 효과적으로 지원할 수 있는 솔루션을 개발하였다. 개발된 솔루션은 블루투스 표준안에 정의된 핵심 스택 계층과 오디오 응용을 위해 제안된 관련 프로파일들을 SoC 기반 플랫폼에 완전히 임베딩된 형태로 구현하였고, 타겟 SoC에서 지원하는 기능들을 이용하여 다양한 최적화 기법들을 적용하였다.

본 논문의 구성은 다음과 같다. 2장에서는 임베디드 블루투스 오디오 스트리밍 솔루션 개발을 위한 개발 환경에 대해 살펴 본다. 3장에서는 본 논문에서 개발한 솔루션의 핵심 기능인 임베디드 블루투스 프로토콜 스택 및 프로파일의 구현 내용에 대해 설명한다. 4장에서는 개발 환경인 DSP 코어 내장 SoC의 기능을 이용하여 오디오 재생 성능을 최적화하기 위한 기법들을 소개한다. 5장에서 개발된 솔루션의 성능을 평가하고, 마지막으로 6장에서 결론을 내리고 마치도록 한다.

2. 개발 환경

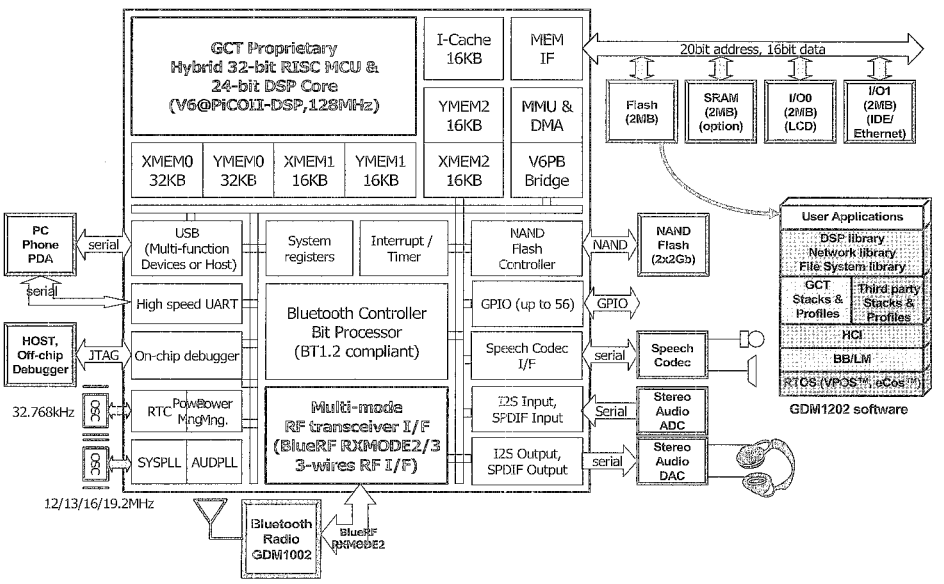
본 논문에서는 개발환경으로 지씨티 세미컨덕터 사의 GDM1202 개발 보드를 사용하였다. GDM1202는 Vincent PiCOII-DSP (P2D) 아키텍쳐[5,6]에 기반한 32비트 RISC 프로세서로 블루투스 베이스밴드 기능을 내장하고 있으며, 멀티미디어 응용 지원을 위한 부가적인 DSP 명령어 집합을 제공한다. 또한, 불필요한 외부 메모리 접근을 막음으로써 시스템의 성능을 향상시키고 전력 소모를 감소시킬 수 있도록 16KB 명령어 캐쉬 (I-Cache) 외에 동적 재구성이 가능한 128 KB의 SRAM 스크래치 패드 (Scratch pad) 메모리를 내장하고 있다. (그림 1)에서와 같이 스크래치 패드 메모리는 X/YMEM0, X/YMEM1, X/YMEM2 등 6개의 영역으로 나누어져 있다. 이 중 X/YMEM2의 경우 블루투스 베이스밴드와 USB 장치가 사용하는 버퍼 영역으로 할당되며, 나머지 영역 (96 KB)은 고속의 명령어/데이터 캐쉬처럼 사용할 수 있으므로 접근 빈도가 높거나 성능영향이 큰 코드나 데이터를 할당하여 성능을 향상시킬 수 있다. 또한 GDM1202는 유닉스 계열 호스트와 윈도우 계열 호스트에서 동작하는 GNU 기반의 교차 개발 환경 (Cross development environment)인 Vincent-sdt 패키지를 제공하고 있다.

운영체제로는 지씨티 세미컨덕터 사에서 자체 개발한 경량 실시간 운영체제인 VPOS (Vincent Proprietary Operating System)를 사용하였다. VPOS는 가상기억장치가 없는 시스템에서 효율적인 메모리 관리와 동기화 프리미티브, 다중 쓰레드(Thread), 블루투스 클럭을 이용한 세밀한 수준의 실시간 쓰레드 스케쥴링을 지원한다. 그 외에도 USB, UART, PCM 코덱, 오디오 코덱 등 주요 주변기기를 위한 디바이스 드라이버를 제공한다.

3. 임베디드 블루투스 프로토콜 스택 및 프로파일 구현

3.1 가상 운영체제: GVOS (GCT Virtual Operating System)

본 논문에서 다루는 임베디드 블루투스 프로토콜 스택과 프로파일 계층은 GVOS라는 가상 운영체제 계층을 기반으로 C 언어로 구현되었다. GVOS는 운영체제 추상화 계층 (OS abstraction layer)을 두어 서로 상이한 운영체제 환경 사이에 큰 수정 없이 블루투스 프로토콜 스택, 프로파일 부분을 이식할 수 있도록 해 주며, 실시간 디버깅이 어려운 임베디드 시스템의 특성을 고려하여 호스트 운영체제에서 소프트웨어를 사전에 검증할 수 있는 환경을 제공한다. 현재 GVOS는 Windows, Linux, VPOS 등의 운영체제에 이식되어 있다. GVOS에서 제공하는 주요 기능들은 <표 1>과 같다.



(그림 1) GDM1202 내부 구조

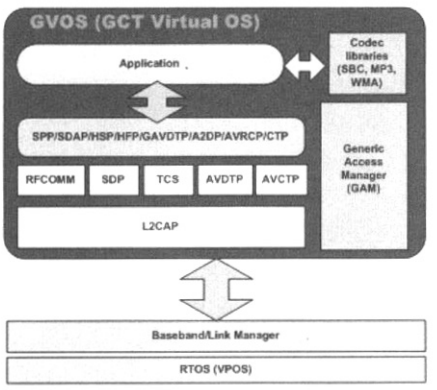
〈표 1〉 GVOS 주요 기능

주요 기능	상세 설명
태스크 관리	◦ 태스크 단위(블루투스 스택, 프로파일 계층의 기본 구현단위)로 스케쥴링 ◦ 사건 기반 (event-driven) 스케쥴링 방식으로 동작
메모리 관리	◦ 정적 최대 크기를 가진 메모리 풀 내에서의 동적 할당 방식 ◦ 시스템 상황을 고려한 버퍼 풀 크기 조정 가능 ◦ 메시지 버퍼 풀과 데이터 버퍼 풀 지원 ◦ ACL 데이터 처리의 효율성을 위한 계층간 Zero-copy 전송
타이머 관리	◦ 블루투스 스택, 프로파일 계층과 응용 프로그램을 위한 복수 개의 타이머 지원 ◦ 타이머 단위: 250 ms (블루투스 클럭 단위 625 us 기반하여 조정 가능)

3.2 임베디드 블루투스 프로토콜 스택 및 프로파일

일반적으로 블루투스 프로토콜 스택 계층은 L2CAP, RFCOMM, SDP 등을 포함하며 이들을 핵심 (Core) 스택 계층이라고 한다[7]. 그리고, 각각의 응용에 따라 특화된 블루투스 응용 레벨 프로토콜들은 프로파일(Profile)이라 불린다. 본 논문에서 구현한 임베디드 블루투스 프로토콜 스택/프로파일 솔루션은 (그림 2)와 같은 프로토콜 계층과 프로파일들을 포함한다. 이들 중 오디오 스트리밍에 관계되는 계층은 다음과 같다[8~12].

◦ AVDTP (Audio/Video Distribution Transport Protocol): 오디오/비디오 데이터 전송을 위한 점대점 (point-to-point) 제어 정보를 교환하는 데 사용되는 프로토콜



(그림 2) 임베디드 블루투스 솔루션 계층구조

◦ AVCTP (Audio/Video Control Transport Protocol): 오디오/비디오 데이터 스트림의 재생/일시중지/볼륨 조정 등의 특성에 대한 원격 제어를 위한 프로토콜

◦ GAVDP (Generic Audio/Video Distribution Profile): 오디오/비디오 스트리밍을 위한 ACL 채널 관리를 위한 일반적인 사항을 정의하는 프로파일

◦ A2DP (Advanced Audio Distribution Profile): AVDTP 프로토콜 상에서 ACL 채널을 통한 오디오 스트리밍 응용의 동작을 명세하는 프로파일

◦ AVRCP (Audio/Video Remote Control Profile): AVCTP 프로토콜 상에서 ACL 채널을 통한 오디오/비디오 채널 제어 호환성을 명세하는 프로파일

본 논문에서 구현한 각 계층의 주요 API 함수는 <표 2>와 같다.

〈표 2〉 임베디드 A/V 프로파일의 주요 API 함수

계층	함수명	설명
A2DP	A2DP_Config()	A2DP 프로파일 구성 설정 (오디오 데이터 송신/수신 핸들러 등록)
	A2DP_Stream()	오디오 데이터 송신 함수
	A2DP_Open(), A2DP_Close() A2DP_StartStream(), A2DP_SuspendStream() A2DP_ReConfigStream()	A2DP 오디오 스트림 관리 함수 (연결 설정/해제, 스트림 시작/중지, 스트림 속성 재지정)
AVDTP	AVDT_RegisterCallbacks()	AVDTP->A2DP 이벤트 처리 함수 등록
	AVDT_RegisterSEP()	스트림 엔드 포인트 (SEP) 등록
	AVDT_ConnectReq(),AVDT_ConnectRsp(), AVDT_DisconnectReq(), AVDT_DisconnectRsp(), AVDT_OpenStreamReq(), AVDT_OpenStreamRsp(), AVDT_CloseStreamReq(), AVDT_CloseStreamRsp() AVDT_StartStreamReq(), AVDT_StartStreamRsp() AVDT_SuspendStreamReq(), AVDT_SuspendStreamRsp()	AVDTP 트랜잭션 처리 함수 (기본 미디어 트랜스포트 기능 구현, SEP 탐색, 코덱 정보 중재, 보안데이터 전송 기능 포함)
AVRCP	AVRCP_Config()	AVRCP 프로파일 구성 설정
	AVRCP_Connect(), AVRCP_Disconnect()	AVRCP 연결 설정/해제
	AVRCP_UnitInfoCmd(), AVRCP_SubunitInfoCmd(), AVRCP_PassThroughCmd()., AVRCP_PassThroughCmd(), AVRCP_SendGenericAVC()	리모트 컨트롤 명령어 전송
AVCTP	AVCT_RegisterCallbacks()	AVCTP->AVRCP 이벤트 처리 함수 등록
	AVCT_ConnectReq(), AVCT_ConnectRsp() AVCTP_DisconnectReq(), AVCT_SendMessage()	AVCTP 연결 설정/해제 및 메시지 전송 함수

또한, 본 논문에서 제안한 임베디드 블루투스 오디오 스트리밍 솔루션은 블루투스 오디오 표준 코덱인 SBC (SubBand Codec) 뿐만이 아니라 MP3, WMA 등 다양한 오디오 재생 장치에서 널리 쓰이는 코덱을 지원함으로써 그 적용 범위를 넓힐 수 있도록 하였다.

3.3 오디오 스트리밍 구조

본 논문에서 개발한 오디오 스트리밍 솔루션의 동작은 (그림 3)과 같다. (그림 3)은 SBC 코덱에 기반한 오디오 스트리밍 환경이다. 오디오 송신 장치는 노트북이나 PDA와 같은 음원 장치로부터 PCM 데이터 입력을 받아 SBC 코덱을 이용한 실시간 인코딩을 수행하고, 인코딩된 데이터는 무선 전송을 위한 공유 버퍼인 audenc_buf에 저장된다. 블루투스 A2DP 프로파일 계층에서 수행되는 A2DP_Stream() 함수가 audenc_buf의 내용을 주기적으로 검사하여 SBC 프레임 크기의 정수배만큼 데이터가 확보되면 AVDTP 계층을 통해 수신 장치로 데이터를 전송하게 된다. 병행 수행을 위해 SBC 인코더 엔진과 블루투스 프로파일/프로토콜 계층은 별도의 VPOS 쓰레드로 구현하였으며, 공유 버퍼인 audenc_buf에 대한 상호배제는 VPOS에서 제공되는 mutex 프리미티브를 이용하여 구현하였다. 오디오 수신 장치에서는 SBC 데이터 수신시 비동기적으로 불리는 데이터 핸들러를 통해 SBC 디코더와 공유하는 auddec_buf에 데이터를 저장하게 된다. 오디오 송신장치와 마찬가지로, 별도의 VPOS 쓰레드로 동작하는 SBC 디코더 엔진은 주기적으로 auddec_buf를 검사하여 1개 이상의 SBC 프레임의 수신이 확인되면 디코딩 작업을 수행한다.

4. 오디오 스트리밍 최적화 기법

4.1 스크래치 패드 메모리의 활용

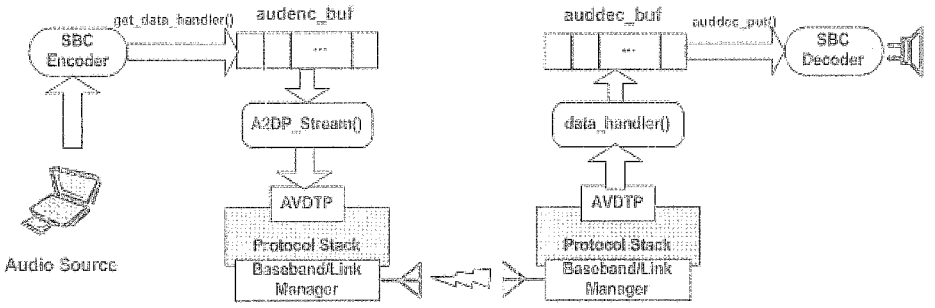
본 논문에서는 개발 플랫폼으로 사용한 GDM1202의 스크래치 패드 메모리를 활용하여 성능 향상과 전력 소모 절감 효과를 거두도록 하였다. 스크래치 패드 메모리는 2장에서 설명한 바와 같이 6개의 영역으로 구분되며, 블루투스 베이스밴드/USB 버퍼로 할당되어 있는 영역을 제외한 나머지 영역은 명령어/데이터 캐쉬처럼 사용할 수 있다.

스크래치 패드 메모리의 활용에 있어서 첫째 단계는 사용 빈도와 크기에 따라 스크래치 패드 메모리에 적재할 함수와 데이터를 선정하는 작업이다. 함수 선정 작업은 Vincent 교차 개발 환경에서 제공하는 시뮬레이터의 프로파일링 기능을 사용하였다. Vincent 시뮬레이터는 하드웨어 환경 설정 파일과 실행 이미지를 입력으로 하며, 프로파일링 기능을 작동시킬 경우 전체 수행시간 동안 각 함수가 호출된 횟수와 함수 수행시간, 함수 크기 등을 출력으로 얻을 수 있다. 일례로 블루투스 오디오 전송에서 표준으로 사용되는 SBC 인코더와 디코더에서 사용되는 함수들의 특성을 프로파일링을 통해 추출한 결과는 <표 3>과 같다.

이와 같은 프로파일을 바탕으로 스크래치 패드 메모리의 명령어 영역에 적재할 함수를 결정할 때의 우선순위 설정은 전체 수행시간에서 차지하는 비중, 함수 크기의 순서를 따른다. 먼저 전체 수행시간에서 차지하는 비중이 높은 함수를 우선적으로 선정하고, 수행시간의 차이가 일정 비율 이하일 경우 크기가 작은 함수를 우선적으로 선정하여 더 많은 함수가 스크래치 패드 메모리에 적재될 수 있도록 한다. 함수의 호출 횟수가 아니라 수행시간을 기준으로 대상 함수를 선정하는 이유는 수행시간, 즉 함수 내의 명령어 수가 함수의 호출 횟수보다 전력 소모에 미치는 영향이 더 크기 때문이다. 데이터의 경우에는 전체 소프트웨어의 각 부분에서 정적으로 할당하여 사용하는 버퍼들을 먼저 우선적으로 선정하도록 한다. 이러한 버퍼로는 GVOS 데이터, 메시지 버퍼, 오디오 데이터 버퍼 등을 들 수 있다. 일반적으로 이러한 버퍼의 크기가 각각 1 KB~16 KB 정도로 전체 스크래치 패드 메모리의 크기에 비

〈표 3〉 SBC 디코더의 함수 프로파일 요약 (전체 시뮬레이션 시간: 45,968,757 cycle)

함수명	호출횟수	수행시간 (cycle)	크기 (bytes)
sbcd_subband_synthesis_filter8	346	7000964	452
sbcd_write_pcm	346	3910838	380
sbcd_getbits	13578	2257745	356
sbc_decode_refill_buffer	161	620669	192
sbcd_reconstruct_subband_samples	346	455274	280
sbc_decode_header	346	50516	628
sbc_decode_frame	346	26988	332
sbcd_core	347	16628	296



(그림 3) 오디오 스트리밍 구조도

〈표 4〉 GDM1202의 스크래치 패드 메모리 적재 방법

적재방식	메모리 지정 방법

링커 스크립트 지정	스크래치 패드 메모리 영역 속성 지정: VGROUP directive 사용
	# VGROUP(name, attribute, align, offset, grouping rule, size limit, attach location)
	VGROUP(YMEM0, 0, 6, 0x0000, 0x0000, 0xA, 0x8000)
	VGROUP(YMEM0_MP3, 0, 6, 0x0000, 0x0000, 0xA, 0x8000)
	VGROUP(YMEM0_WMA, 0, 6, 0x0000, 0x0000, 0xA, 0x8000)

	영역에 들어갈 심볼 지정: VSECTION directive 사용 VSECTION(_sbcdec_info, YMEM0) VSECTION(_mp3d_info, YMEM0_MP3) VSECTION(_wmadec_info, YMEM0_WMA)
함수/변수 속성 지정	함수 지정 예 void critical_func(int) **text_xmem1**;
	변수 지정 예 int critical_data **data_xmem0** = 0;

해 큰 부분을 차지하므로 그 이외의 일반 변수가 스크래치 패드 메모리에 적재될 여지는 매우 낮다.

이와 같이 선정된 함수와 데이터를 특정 스크래치 패드 메모리 영역에 적재하도록 지정하는 방법은 <표 4>와 같다. 첫째, 링커 스크립트에서 각 영역의 전체 혹은 일부를 명령어 혹은 데이터 영역을 쓸 것인지를 지정하고 이 영역에 적재할 함수 혹은 데이터 심볼을 등록할 수 있다. 등록된 함수 혹은 데이터 심볼은 VPOS의 초기화 단계에서 해당 스크래치 패드 메모리 영역에 복사된다. 둘째, 데이터, 즉 변수의 경우는 필요에 따라 프로그램 코드 내에서 속성 지정을 이용해 스크래치 패드 메모리에 적재함을 명시하고 적재할 영역을 설정할 수 있다. 동일한 심볼이 링커 스트립트와 속성 지정을 통해 각각 다른 영역에 할당될 경우에는 링커 스크립트의 영역 지정이 우선적으로 사용된다.

<표 4>의 예를 보면 YMEM0 영역에 서로 다른 변수들을 중첩(overlay)해서 적재할 수 있는데, 이 기능은 응용 프로그램의 특성상 특정 시간에 배타적으로 쓰이는 데이터나 함수 집합이 존재할 때 사용된다. 예를 들어, 본 논문에서 개발한 솔루션은 오디오 스트리밍 응용 프로그램의 경우 오디오 파일의 압축형태에 따라 데이터 재생시 사용되는 데이터와 함수가 달라진다는 점을 활용하여 응용 프로그램에서 재생을 위한 코덱이 변경되었음을 인식한 시점에 동적으로 해당 스크래치 패드 메모리 영역에 필요한 코덱의 정보를 복사하도록 한다. 이와 같은 기법을 적용하면 추가 메모리의 확장 없이 복수의 코덱을 지원할 수 있다.

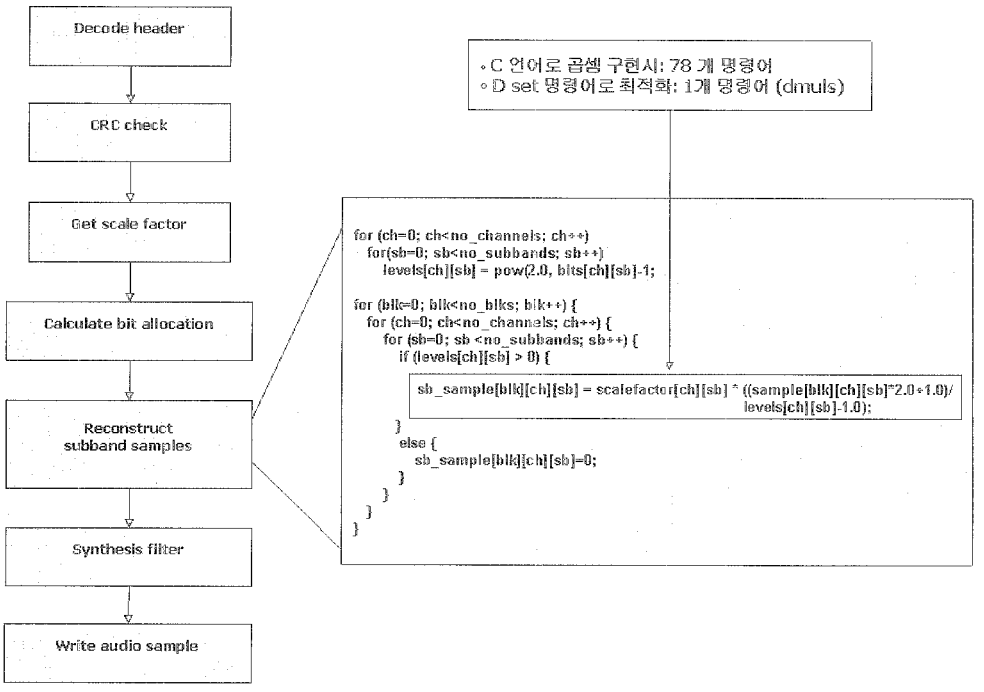
4.2 코덱 최적화

본 연구에서는 A2DP 프로파일에서 사용되는 주요 코덱 라이브러리의 핵심 부분을 Vincent 아키텍처에서 제공하는 DSP 명령어, 블록단위 데이터 처리 명령어를 이용한 어셈블리 코드로 구현하여 성능을 향상하도록 하였다. <표 5>에 요약된 GDM1202 명령어 집합 중 D set과 M set에 속하는 명령어들이 코덱 최적화에 사용되었다. 일반 RISC 프로세서 환경에서는 코덱에서 빈번히 사용되는 연산들을 C 언어로 구현하거나 별도의 DSP 프로세서를 사용해서 구현하여야 하지만, GDM1202 플랫폼에서는 D set 에서 지원하는 DSP 확장 명령어들을 이용하여 이러한 부담을 제거하였다. (그림 4)에서는 SBC 디코딩 과정의 각 단계를 보여 주고 있는데, 이 중 서브밴드 샘플을 재구성하는 과정에서 그림에서 보는 바와 같이 C 언어로 고정 소수점 연산을 구현할 때는 78개의 어셈블리 명령어를 사용하여야 한다. 그러나, 이 부분은 GDM1202가 제공하는 DSP 확장 명령어 중 하나인 "dmuls" 명령어 하나로 대치해서 사용할 수 있다.

또한, 스크래치 패드 메모리 활용을 최대화하기 위해 병렬 메모리 읽기 기능을 지원하는 M set 명령어도 성능 향상에 도움을 준다. M set에 속하는 명령어들은 XMEM 혹은 YMEM과 같은 내부 스크래치 패드 메모리 접근과 동시에 데이터 처리 명령을 수행할 수 있도록 설계되었다. 예를 들어 동시에 연산 처리에 사용될 수 있는 코덱 관련 연산 테이블들을 각각 XMEM과 YMEM에 저장해 두고 <표 5>의 명령어 중 M set에 속하는 ldx, ldy 명령어를 사용하여 병렬적으로 데이터를 접근하도록 구현할 수 있다. 본 연구

〈표 5〉 Vincent 아키텍처 명령어 집합

명령어집합	설명	주요 명령어 (군)
G set	기본적인 RISC 동작에 관련된 명령어 집합 (단일/다중 데이터 처리, 데이터 전송, 프로세서 제어)	add, sub, mul, and, or, xor, padd, pmul, ppack 등
S set	G set의 축약 버전 (G set 명령어와 1:1 대응)	add, sub, sll, slr, mv, ldw, stw 등
D set	DSP 응용 최적화를 위한 명령어 집합	dmul, dpac, dmixh 등
M set	병렬 메모리 접근 지원	ldx, ldy, stx, sty 등

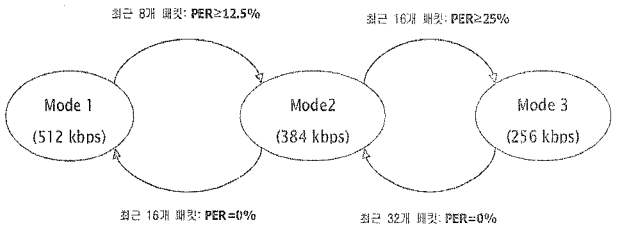


(그림 4) SBC 디코딩 과정과 DSP 명령을 이용한 최적화 예

에서는 SBC 코덱의 인코더, 디코더 구현 부분 중 합성 필터(Synthesis Filter) 처리 부분에서 M set 명령어들을 사용하였으며, 이 경우 M set 명령어를 사용하지 않았을 경우와 대비했을 때의 성능향상 효과는 5장 성능평가 부분에서 자세히 다루도록 한다.

4.3 동적 오디오 품질 조정

AVDTP 표준에서는 기본 미디어 전송 서비스, 보고 서비스, 복구 서비스를 정의하고 있으나 본 논문에서 개발된 AVDTP 프로토콜에서는 기본 미디어 전송 서비스만을 제공하고 있다. 이 점을 보완하기 위해 본 연구에서는 간섭 요소가 많은 무선 통신 환경을 고려하여 주파수 대역이 겹치는 타 무선 장치나 근거리의 다른 블루투스 장치의 간섭 효과에 의해 통신 환경이 나빠질 경우에는 오디오 데이터 송신 품질을 조정하도록 하였다. 통신 링크의 상태는 패킷 오류율 (PER: Packet Error Rate)을 기준으로 하였으며, 오디오 송신 장치에서 주기적으로 송신 패킷 오류율을 검사하여 SBC 인코딩 샘플 비트율 (sample bitrate)을 동적으로 변화시킨다[13]. (그림 5)는 3단계로 구성된 품질 조절



(그림 5) 동적 오디오 품질 조정 알고리즘

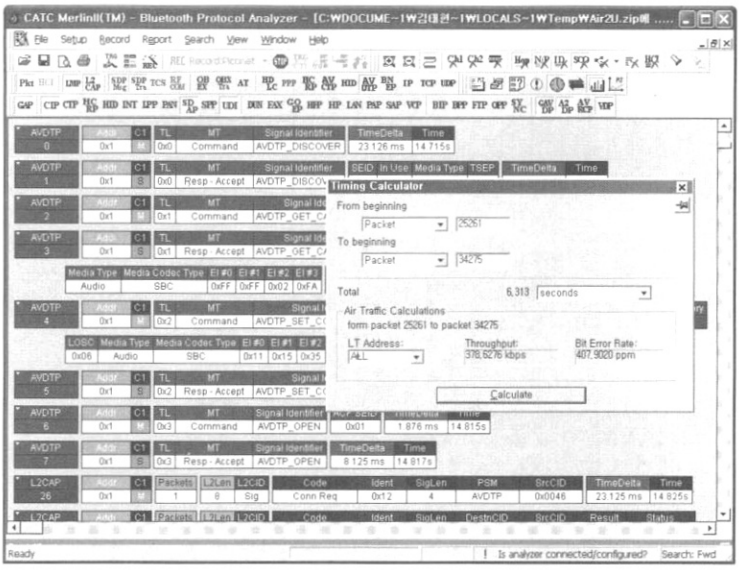
알고리즘을 나타내고 있으며, 다양한 통신 링크 상태에 대응하기 위해서는 더 세분화된 품질 모드를 쉽게 추가할 수 있다.

5. 성능평가

개발된 오디오 스트리밍 솔루션의 성능을 평가하기 위하여 다음과 같은 환경에서 성능을 측정하였다. 오디오 송신 장치는 블루투스 USB 동글과 IVT 사의 BlueSoleil 프로토콜 스택[14]을 장착한 Pentium 4 데스크탑 컴퓨터를 사용하였으며, 오디오 수신 장치는 본 연구에서 개발한 임베디드 블루투스 스택과 오디오 코덱을 내장한 GDM1202 프로토타입 보드를 사용하였다. 성능 평가는 오디오 수신부에서 설정한 재생 주기가 오디오 송신부에서 모든 오디오 프레임 재생시 엄밀하게 지켜지는지를 평가하도록 하였다. 또한, 개발된 오디오 솔루션이 AVDTP 표준에서 정의한 트랜잭션을 정확하게 수행함을 검증하기 위해 Lecroy 사의 블루투스 프로토콜 분석기인 BT Tracer [15]를 사용하였다.

실험에서 사용된 주요 파라미터는 <표 6>과 같다. 오디오 데이터 전송을 위한 ACL 링크의 패킷 타입은 DH5로 설정하였으며, 코덱은 SBC를 사용하였다. 실제 전송에 사용되는 코덱 파라미터는 오디오 송신 장치에서 설정되므로 수신 장치에서 SBC 프레임 헤더를 해석하여 추출하였다. 본 논문에서 기준으로 제시한 오디오 재생율은 블루투스 표준안에 명시된 고품질 오디오의 범주에 속한다[11].

실험 내용과 결과는 다음과 같다. 첫째, 개발된 블루투스 오디오 스트리밍 솔루션이 블루투스 AVDTP 표준안을 충실히 수행하는지를 검증하기 위해 IVT BlueSoleil 과의 오



(그림 6) AVDTP 트랜잭션 동작 (프로토콜 분석기 캡쳐 화면)

〈표 6〉 실험 파라미터

설정 내용	설정값
ACL 패킷 타입	DH5 (MTU 크기: 335 바이트)
코덱 종류	SBC
코딩 파라미터	샘플링 주기(Sampling rate): 48 kHz 재생율(Bitrate): 357 kbps, 스테레오 프레임 길이: 119 바이트 프레임 내 블록 수: 16 블록 내 서브밴드 수: 8

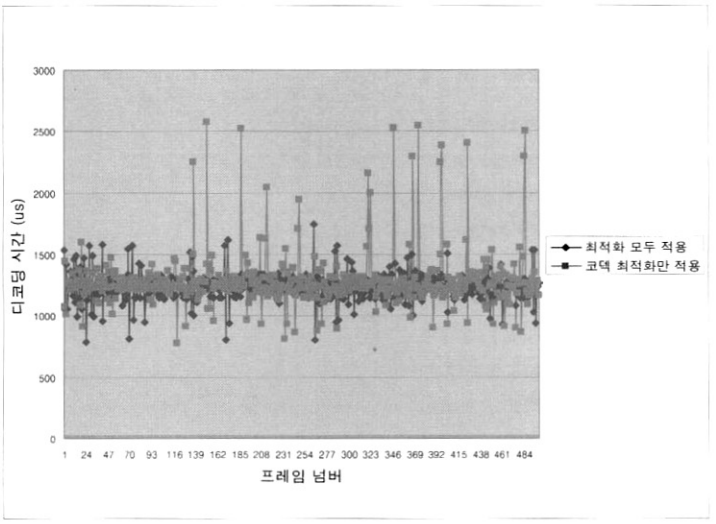
〈표 7〉 오디오 재생 시간 측정 결과 특성 (기준 시간: 2667 us)

최적화 정도	오디오 프레임 재생 시간 (us)		
	평균	최대	최소
최적화 적용 없음	39802	44520	37560
메모리 최적화만 적용	34831	37808	33440
코덱 최적화만 적용	1278	2576	776
모든 최적화 적용	1233	1744	784

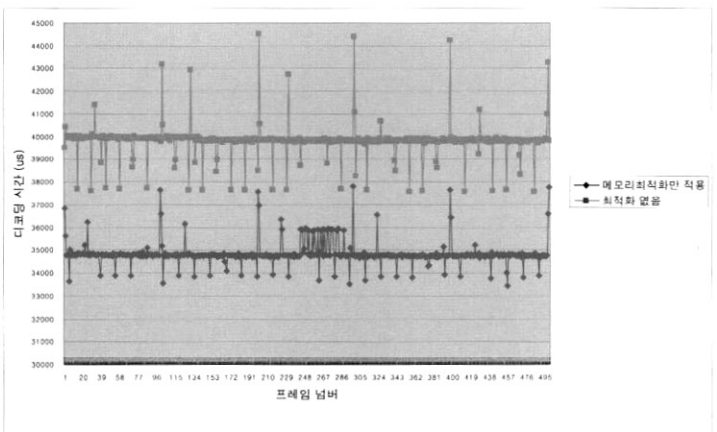
디오 스트리밍 과정을 프로토콜 해석기로 캡쳐하였다. (그림 6)을 보면 개발된 솔루션은 블루투스 표준에 명시된 서비스 설정 과정을 잘 수행함을 알 수 있다. 둘째, 적용된 최적화 기법들이 성능을 미치는 영향을 분석하기 위해 GDM1202에서 제공되는 타이머를 이용하여 SBC 코덱으로 인코딩된 오디오 프레임의 재생시간을 오디오 수신부에서 측정하였다. 사용된 타이머의 기준 단위는 8 us이다. <표 7>의 파라미터에 따른 오디오 한 프레임의 재생 주기는 다음과 같다. 하나의 오디오 프레임은 프레임 내 블록 수와 블록 내 서브밴드 수의 곱에 해당하는 개수의 오디오 샘플로 구성되므로, 본 실험에서는 한 오디오 프레임 내에 128개의 오디오 샘플이 저장된다. 매 1/48000 초마다 하나의 오디오 샘플이 샘플링되므로, 한 프레임의 재생 주기는 128/48000 (seconds) = 약 2667 us가 된다. 즉, 오디오 수신부에서 오디오 재생 시간을 측정했을 때 매 오디오 프레임의 재생이 2667 us 이내에 완료될 경우 송신부의 오디오 품질이 유지된다고 할 수 있다.

(그림 7)은 최적화 수준을 달리하며 오디오 수신 장치에서 하나의 오디오 프레임을 재생하는 데 걸리는 시간을 500 프레임에 대해 측정한 결과를 보여 준다. (그림 7-a)는 최적화 기법을 모두 적용한 경우와 코덱 최적화만 적용한 경우, (그림 7-b)는 메모리 최적화만 적용한 경우와 최적화를 적용하지 않은 경우의 측정 결과를 각각 나타내고 있다. <표 7>은 (그림 7)의 측정 결과를 요약한 내용을 보여 준다. (그림 7)과 <표 7>의 결과에서 알 수 있듯이, 최적화 기법을 모두 적용했을 경우 평균적으로 기준 시간의 46% 내에 오디오 프레임의 해석 및 재생이 완료됨을 알 수 있고 최대 소요시간의 경우를 살펴 보더라도 기준 시간의 66% 이내에 완료됨을 알 수 있다. 실험에서 주목할 만한 점은 코덱 최적화만 적용하더라도 모든 오디오 프레임이 송신부의 품질을 유지하면서 재생됨을 알 수 있다. 그러나, 이 경우 최대 재생시간이 기준 시간의 97%에 근접하므로 주변 통신 환경의 악화로 인한 재전송 처리에 걸리는 시간 오버헤드가 늘어날 경우 재생 주기를 만족하지 못 할 가능성을 안고 있다. 또한, 내부 스크래치 패드 메모리의 활용도가 낮아지므로 외부 메모리의 접근 빈도가 높아짐에 따라 전력 소모가 늘어날 수밖에 없다. 메모리 최적화만 적용할 경우와 최적화를 적용하지 않은 경우 평균적으로 기준 재생 시간의 13~15배에 해당되는 시간을 소모하므로 오디오 스트리밍에 사용하기는 부적절하다고 할 수 있다.

마지막으로 GDM1202의 병렬 메모리 접근을 지원하는 M set 명령어 사용에 따른 성능 향상 효과를 측정하였다. 4.2절에서 설명한 바와 같이 M set 명령어는 SBC 코덱의 경우 합성 필터 처리부분에서 사용되었다. 합성 필터 처리 함

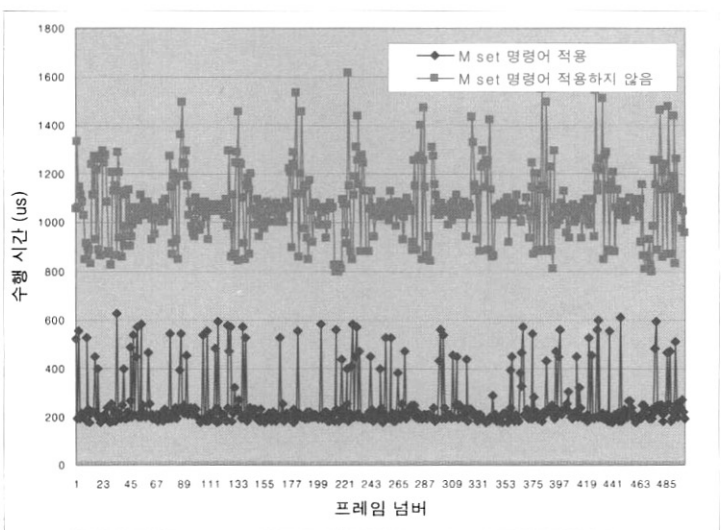


(a) 최적화를 모두 적용한 경우와 코덱 최적화만 적용한 경우



(b) 메모리 최적화만 적용한 경우와 최적화를 적용하지 않은 경우

(그림 7) 최적화 수준에 따른 오디오 프레임 재생시간 측정 결과 (전체 500 frame)



(그림 8) 병렬 메모리 접근 명령어의 성능 향상 효과

수에서 M set 명령어의 영향만을 분석하기 위해 DSP 명령어로 구현한 부분들은 동일하게 사용하도록 하고 필터 연산에 사용되는 데이터도 내부 메모리에 저장한 상태에서, M set 명령어를 사용하였을 경우와 C 언어로 데이터 접근 부분을 구현하였을 경우에 대해 해당 함수의 수행 시간을 측정, 비교하였다. (그림 8)의 결과를 보면 M set 명령어로 최적화를 수행한 경우 평균적으로 4.25배(1065 us/251 us) 정도의 성능 향상을 보이는 것으로 나타났다. 이 결과는 M set 명령어를 사용할 경우 별도의 베이스 주소 레지스터를 사용하는 데 따른 범용 레지스터 파일에 대한 접근 충돌의 감소, 독립적인 입출력 포트를 가진 X/YMEM에 저장되어 있는 데이터들에 대한 병렬적 접근으로 인한 메모리 접근 오버헤드 감소, 데이터 접근 명령어와 데이터 처리 명령의 병렬적 수행으로 인한 성능 향상이 중첩된 결과로 볼 수 있다.

6. 결 론

본 논문에서는 향후 블루투스 기술을 이용하여 가장 널리 쓰일 것으로 기대되는 오디오 스트리밍 솔루션을 개발하였다. 개발된 솔루션은 다음과 같은 특성을 가진다. 첫째, 범용 PC 응용과 임베디드 기기 등 다양한 플랫폼에 빠른 시간 내에 쉽게 적용할 수 있도록 가상 운영체제 계층을 두어 이식성을 향상시켰다. 둘째, 최근에 와서 그 적용 범위가 확대되고 있는 개인 휴대용 멀티미디어 정보기기에 대응할 수 있도록 블루투스 프로토콜 스택과 프로파일을 SoC 칩 내에 집적하였다. 셋째, 개발 플랫폼에서 제공하는 DSP 명령어를 사용하여 코덱 라이브러리를 최적화함으로써 전력 소모 최소화와 성능 향상을 이룰 수 있다. 또한, 블루투스 표준에서 기준이 되는 SBC 코덱 외에도 기존 오디오 응용에서 많이 사용되고 있는 MP3, WMA 등의 코덱을 지원함으로써 솔루션의 적용 범위를 넓힐 수 있었다. 넷째, 자주 쓰이는 프로그램 코드와 데이터를 SoC 칩에 통합된 동적 재지정 가능 스크래치 패드 메모리를 적재함으로써 불필요한 외부 메모리 접근을 막아 전력 소모를 감소시키고 성능을 향상시킬 수 있으며, 필요에 따라 코덱 라이브러리를 동적으로 치환함으로써 다양한 코덱의 사용을 쉽게 하였다. 다섯째, 다양한 무선 통신 환경을 고려한 동적 품질 제어 기법을 적용하였다.

본 논문에서 개발된 솔루션은 PC와 스테레오 헤드셋, 홈 씨어터 시스템에서의 DVD 재생기와스피커 시스템, 혹은 MP3 플레이어와 스테레오 헤드셋의 연동 등의 응용에 적용할 수 있으며, 휴대 전화 응용에서 필요로 하는 헤드셋/핸즈프리 프로파일을 통합, 지원함으로써 휴대 전화기와 스테레오 헤드셋을 연동하는 응용으로까지 확장이 가능하다.

참 고 문 헌

[1] J. Bray and J. F. Sturman, "Bluetooth: Connect Without Cables," Prentice Hall.

[2] S. Zeadally and A. Kumar, "Protocol Support for Audio Streaming between Bluetooth Devices," Proceedings of the 2004 IEEE Radio and Wireless Conference, pp. 303-206, 2004.

[3] P. Bellavista, C. Stefanelli, and M. Tortonesi, "The ubiQoS Middleware for Audio Streaming to Bluetooth Devices," Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, pp. 138-145, 2004.

[4] A. Floros, N-A. Tatlas, and J. Mourjopoulos, "BlueBox: A Cable-free Digital Jukebox for Compressed-quality Audio Delivery," IEEE Transactions on Consumer Electronics, Vol. 51, No.2, pp. 534-539, 2005.

[5] 김정민, "무선 임베디드 애플리케이션을 위한 두개의 동적 VLIW 스테이트를 갖는 멀티미디어 프로세서," 서울대학교 박사학위논문.

[6] H-J. Suh, H. J. Lim, J. Kim, "GDM5302: Wireless Integration of a Mobile Multimedia Processor," Proceedings of the International Conference on Embedded Systems and Applications, pp. 182-187, 2005.

[7] Bluetooth Specification, http://www.bluetooth.org

[8] Bluetooth Special Interest Group, "Audio/Video Distribution Transport Protocol Specification," Version 1.0.

[9] Bluetooth Special Interest Group, "Audio/Video Control Transport Protocol Specification," Version 1.0.

[10] Bluetooth Special Interest Group, "Generic Audio/Video Distribution Profile Specification," Version 1.0.

[11] Bluetooth Special Interest Group, "Advanced Audio Distribution Profile Specification," Version 1.0.

[12] Bluetooth Special Interest Group, "Audio/Video Remote Control Profile Specification," Version 1.0.

[13] Hyo Jin Choi, Jinhwan Jeon, Taehyoun Kim, Hyo-Joong Suh, and Chu Shik Jhon, "Robust Delay Control for Audio Streaming over Wireless Link," IEICE Transactions on Information and Systems, E89-D(8), pp.2448-2451, 2006.

[14] http://www.bluesoleil.com/

[15] http://www.lecroy.com/tm/products/ProtocolAnalyzers/bluetooth.asp

김 태 현

e-mail : thkim@uos.ac.kr

1994년 서울대학교 컴퓨터공학과(학사)

1996년 서울대학교 컴퓨터공학과 (공학석사)

2001년 서울대학교 전기 컴퓨터공학부 (공학박사)

2001년~2005년 지씨티 리써치 책임연구원

2005년~현 재 서울시립대학교 기계정보공학과 조교수

관심분야 : Embedded Systems, Real-Time Systems, Wireless Personal Area Network