TMS320C6201 DSP를 이용한 8 채널 실시간 TTS 구현

최준용, 박익현, 박권원, 안진형
LG 전자 디지털네트워크연구소

Real-time Implementation of a 8 channel TTS Using a TMS320C6201 DSP

Jun-Yong Choi, Ik-Hyun Park, Kwon-Won Park and Jin-Hyung Ahn

Digital Network System Lab., LG Electronics Inc.
E-mail: [jychoi,v108369,mimicro,ajh2000]@lgic.co.kr

요약

본 논문에서는 TTS 알고리듬을 16 비트 고정 소수점 DSP인 TMS320C6201을 이용해 다채널 실시간 구현하였으며, 실제로 음성처리 부가 서비스 시스템에 보드 형태로 구현하여 응용하였다. 구현된 TTS는 최적화 작업을 통해 최대 40 MHz 클록으로 채널 당 2초의 합성음 생성하도록 했으며, 개발된 TTS 보드는 두 개의 DSP를 사용하여 DSP 당 8 채널씩 총 16 채널을 수용하였다. 실험 결과, 모든 채널에서 실시간적으로 음성 합성이 수행됨을 확인하였다.

I. 서론

유무선 전화망을 통하여 서비스되는 통신 시스템중 부가 서비스 시스템은 VMS(Voice Mailing Service), ARS(Automatic Response Service)와 같은 녹음 및 재생 전용 음성 처리 시스템에서 UMS(Unified Messaging Service)와 같은 미디어 변환 및 네트워크 통합 시스템으로 변화하고 있다. 이러한 음성 처리 시스템들에서 최근 공통적으로 요구되는 응용 서비스 중의 하나가 Text 입력을 음성으로 변환하는 TTS(Text-To-Speech)기능이다. TTS는 사람과 기계가 의사 소통하기 위하여 중점 연구되어온 분야로써 전화를 걸어 듣는 131 기상 예보 서비스와 같이 정보 처리의 결과를 문자나 영상이 아닌 음성으로 출력하여 사용자에게 좀더 쉬운 인터페이스를 제공하고자 하는 곳에서 요구되어 왔다. 특히 최근에는 음성 브라우징 및 보이스 포털등 인터넷과 연계된 서비스에서 TTS는 필수적인 요소가 되었다.

TTS는 크게 언어 분석부와 합성부로 구성된다. 본 논문은 언어 분석부에서 규칙 기반한 운율 추출을 사용했고, 합성부에서 TD-PSOLA(Time Domain-Pitch Synchronous OverLap Add[1-3])방식을 사용하였다. 지금까지 TTS 기능은 PC나 워크스테이션과 같은 서버로 포팅(porting)이 되어 상용화 되었다. 본 논문에서는 TTS 기능을 음성처리 부가시스템에 임베디드(embedded) 보드의 형태로 장착하여 사용할 수 있도록 TTS 알고리듬을 16 비트 고정 소수점 DSP인 TMS320C6201[6-9]을 이용해 다채널 실시간 구현한 내용을 기술한다.

서론에 이어서 II 장에서는 TTS의 구조를 설명하고, III 장과 IV 장에서 TTS의 실시간 구현 및 시스템 실시간 테스트를 설명하며, 5 장의 결론으로 끝을 맺는다.

II. TTS의 구조

2.1 TTS 구조.

TTS[1][2]는 정보 시스템의 텍스트를 입력받아 음성으로 변환하는 기술이다. 이러한 TTS는 합성시 생길 수 있는 문제들을 해결하기 위해 일반적으로 그림 1과 같이 자연 언어 처리부와 음성 합성부로 구성된다.

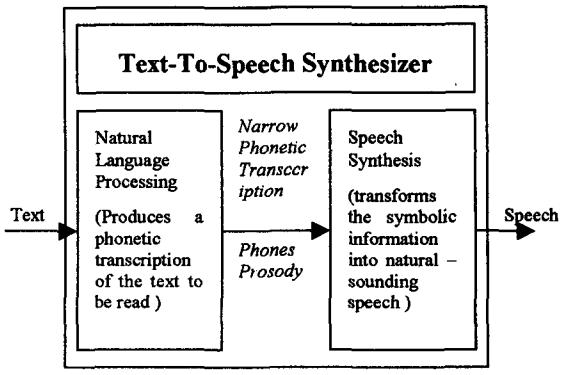

그림 1. 일반적 TTS 구성도[1]

자연 언어 처리부에서는 텍스트를 입력받아 특수한 언어학적인 표현으로 바꾸는 작업을 한다. 이러한 변환은 다음과 같은 세부적인 과정을 거쳐 수행된다[1].

1) 문장 전처리(text preprocessing) : 문장의 처음과 끝 검출 및 텍스트 정규화.
2) 강세 할당(accent assignment) : 영어등 강세가 있는 언어의 경우 강세 수위의 할당.
3) 단어의 발음(word pronounciation) : 발음 규칙 및 예외 발음 규칙에 따라 단어의 발음 할당.

4) 형태소 분석(tagging text) : 검출된 문장내 단어의 형태소 분석.
5) 억양 분리(intonational phrasing) : 텍스트를 몇 개의 억양으로 분리.
6) 음편 지속 시간(segmental durations)결정 : 언어학적인 규칙에 따라 음소의 지속시간 결정.
7) 피치 궤적 계산(pitch contour computation).

음성 합성부에서는 이들 정보를 바탕으로 음성 파형을 합성하는데 아래와 같은 과정을 거쳐 수행된다[1].
1) 주어진 음소에 대한 음성 단위 선택(unit selection).
2) 단위 연결(unit concatenation).
3) 합성 모델에 따른 음성 파형 합성(speech synthesis).

2.2 TTS 소프트웨어 구조

본 논문의 TTS 소프트웨어[4][5][6]는 그림 2 와 같은 블록으로 구성되어 있다. 합성하고자 하는 텍스트가 입력되면 문장 전처리 및 구문 분석부에서 입력 텍스트를 문장으로 분해하여 문장 단위로 처리를 하게 된다. 문장 전처리부에서는 한글, 한자, 영어, 특수 문자, 숫자 등의 모든 입력을 한글로 변환하고 어절의 종류를 표시한다. 어휘 사전과 형태소 사전 및 각종 규칙을 기준으로 하여 어절들을 구문 분석하여 어절들의 문장에서의 역할을 알아내고 문장을 구와 절로 나눈다. 구문 분석을 통해 어휘의 불규칙 변환등을 찾고 음운 변동 규칙표와 불규칙 음운 변동 사전을 통해 문자를 소리나는 대로 바꾼다.

운율 처리부에서는 구문 분석 정보와 음소 정보를 참조하여 합성음에 들어갈 피치, 지속시간(duration), 휴지기간(pause, gap, break)등을 생성한다. 피치는 문장 피치 패턴과 규칙을 이용하여 생성하고 길이와 휴지 구간은 측정을 통해 얻어진 통계치를 이용한다. 그리고, 억양구 읽기 단위(break) 위치 및 종류는 형태소 분석 결과 및 사전에 학습된 HMM(Hidden Markov Model[11])을 이용하여 읽기 단위의 정도를 추정후 결정한다.

합성부에서는 텍스트의 발음 정보에 따라 합성 단위를 생성한다. 생성된 합성 단위에 따라 이미 저장된 합성 DB 에서 합성 단위별 데이터를 가져온다. 이 데이터를 가지고 앞단계에서 생성된 운율 정보에 따라 억양구별로 TD-PSOLA 방식에 의해 피치를 부여하며 합성 파형을 생성한다. TD-PSOLA 는 음질을 향상시키기 위하여 음성 파형의 피치를 필요한 만큼 직접적으로 수정할 수 있는 것이 특징인 음성 합성 기술이다[1].

이들의 블록별 설명은 다음과 같다.
1) 문장 검출 : 전체 text 입력을 문장 단위로 자름.
2) 문장(텍스트) 전처리 : 숫자처리, 부호처리, 날짜 또는 시간처리, 영문 처리, 한자 처리.
3) 발음 변환 : 소리 사전 및 음운 변동 규칙을 이용하여 발음 변환(소리나는 대로 읽기) 수행.
4) 형태소 분석 : 주어진 문장의 형태소 분석 수행.
5) break 결정 : 형태소 분석 결과 및 사전에 학습된 HMM 을 이용하여 각 억양구에 대한 끊어 읽기 위치 및 종류 결정.
6) 피치 정보 초기화 및 태깅 : 피치 패턴을 구하여 각 억양구 별로 태깅.
7) 음소 정보 구하기 : 합성시 사용될 음소 정보를 구함.
8) 음소 및 휴지 구간의 길이 구하기 : 음소 및 음절간 휴지 구간의 길이를 구함.
9) 합성 단위 DB 정보 구하기.
10) 피치값 구하기 : 실제 피치값을 할당.
11) DB data 읽기 : DB 로부터 실제 data 값을 읽음.
12) 억양구 합성 : 얻어진 모든 정보를 이용하여 음성 합성.
13) 어절간 휴지 구간 삽입 : 어절간 적정한 휴지 구간을 삽입.
14) 문장간 휴지 구간 삽입 : 다음 문장과의 휴지 구간을 삽입.

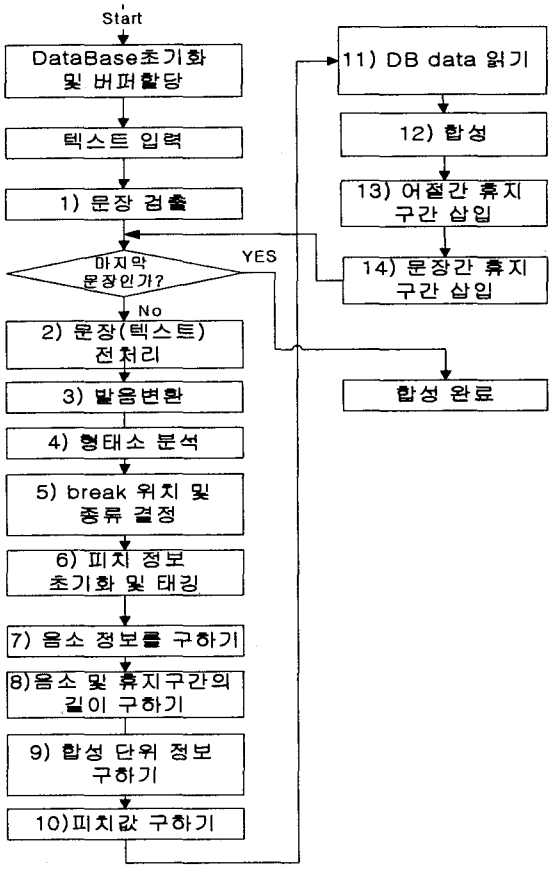

그림 2. TTS 코드의 흐름도.

III. TTS 의 실시간 구현

3.1 TTS 하드웨어 구조

본 논문에서 사용된 Texas Instruments(TI)사의 TMS320C6201 는 16 비트 고정 소수점 DSP 로서 최대 200 MHz(5 ns cycle)로 동작하며 매 사이클마다 최대 8 개의 32 비트의 명령어(instruction)을 수행하여 최대 1600 MIPS(Million Instructions Per Second)의 성능을 갖는다.

메모리 맵은 내부 프로그램 메모리 64 KB, 내부 데이터 메모리 64 KB, 내부 주변장치(internal peripherals) 및 최대 52 MB 의 EMIF(External Memory Interface)에 의한 외부 메모리로 이뤄져 있다.

TTS 의 다채널 실시간 적용을 위하여 그림 3 과 같이 TMS320C6201 가 탑재된 보드를 제작하였다

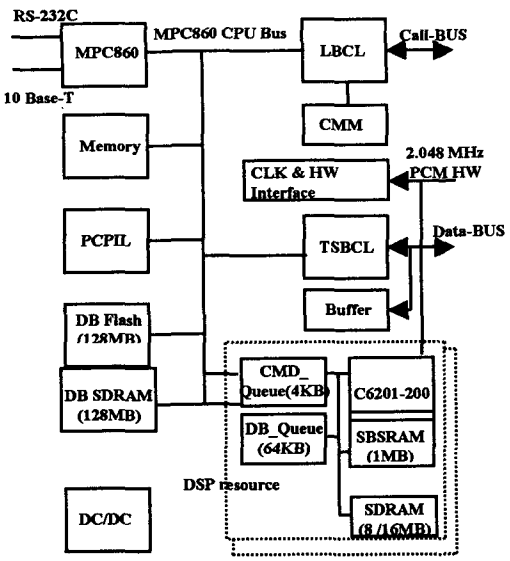

그림 3. TTS 보드 블록도.

TTS 보드의 규격 및 용량은 다음과 같다.

1) 채널용량 : 16 가입자 / 1 보드.
2) Port Processor(PP) : MPC860 66 MHz(32 비트 QUICC Product of Motorola).
3) TTS Processor(TTSP) : TMS320C6201BGJL-200 2 EA
   8 가입자 / TTSP 1 EA.
4) LAN Interface(RJ-45) : CPU 프로그램을 LAN 경유 Flash Memory 에 로딩용.
5) DB Data : LAN 경유 SDRAM 또는 Flash Memory 에 로딩용.
6) DSP part :
   - Clock Rate : 160 MHz.
   - SDRAM : 16 MB * 2 EA .
   - Dual Port RAM : 132 KB * 2 EA.
   - Flash ROM : 128 M B * 1 EA.

TMS320C6201 DSP 는 다양한 부트 구성을 가지고 있는데, 이 부트 구성의 설정은 메모리 맵의 선택, 주소 0x0 의 메모리 형태 선택 및 부트 프로세스의 선택을 포함한다. 본 논문에서는 MAP 1 의 메모리 맵을 사용하고 DSP 의 내부 프로그램 영역을 주소 0 으로 하며 HPI 부트 프로세스를 사용하기 위해 BOOTMODE[4:0] 값을 00111 로 세팅하였다.

DSP 의 TTS 코드는 리셋 후 프로그램 데이터 및 DSP 레지스터의 초기화, 인터럽트 설정, 각 채널에 할당된 버퍼의 초기화를 순서적으로 수행한다. 이렇게 초기화한 후 MPC860 의 시작 명령 및 텍스트를 대기하고 있다가 메시지가 수신되면 실시간적으로 음성 합성한다. 이렇게 합성된 합성음은 인터럽트에 의해 직렬 포트를 통해 시스템의 TDM(Time Division Multiplexer)으로 전달되어 D/A(Digital to Analog) 변환 보드에서 재생된다[11].

3.2 한 채널 실시간 구현

TTS 의 실시간 구현을 위해서 C 소스로 되어 있는 TTS 엔진을 분석한 후 최적화하였다. TMS3320C6201 DSP 는 프로그램 코드가 내부 프로그램 메모리에 존재할 때 최적의 성능을 발휘하지만 본 논문의 TTS 는 코드의 크기 및 사용되는 정보의 양이 방대하여 외부 메모리를 많이 사용하였다. 외부 메모리 사용으로 인한 처리 속도의 하락을 막기 위해 DSP 의 프로그램 모드를 Cache Enable 로 하였고, 빈번한 사용으로 속도에 큰 영향을 주는 변수 및 버퍼를 선별하여 내부 데이터 메모리로 할당하였다. 또, 본 논문에서 사용중인 고정 소수점 DSP 는 부동 소수점 연산에 많은 처리 시간을 필요로 하므로 부동 소수점 변수 할당 및 연산을 최소화하였고, G.711 변환시 테이블 처리하였으며, 외부 메모리의 사용을 최소화하여 처리 시간을 단축하였다.
DSP 의 처리 능력을 최적화하여 실시간적으로 구현된 TTS 의 블록 구성은 그림 4 와 같다.

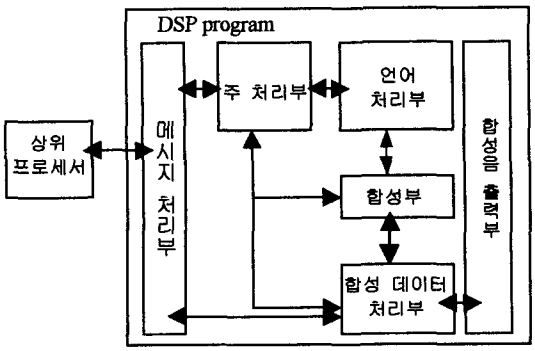

그림 4. DSP 에 구현된 TTS 의 블록도.

주 처리부는 프로그램 초기 구동시 각 변수들을 초기화하고, TTS시작 명령을 수신했을 경우 언어 처리부, 합성부, 합성 데이타 처리부를 호출하여 전체 음성 합성 과정을 조정하는 역할을 한다. 주 처리부와 MPC860간의 메시지 통신을 담당하는 메시지 처리부는 DSP의 타이머에 의해 동작하며, 주 처리부에 의해 호출된 언어 처리부는 문장 단위로 언어 처리를 하고, 합성부로 추출된 데이터를 전달해 합성음을 생성하도록 한다. 이렇게 생성된 합성음은 DSP의 직렬 포트로 전송 후 재생이 되도록 합성음 출력부에 의해 제어 되며, 시스템 저장 장치로 업로드가 되도록 합성 데이터 처리부에 의해 제어된다.

합성부 출력부에서는 재생 버퍼를 2 개를 두어 합성음 재생시 교차적으로 사용하도록 하였다. 이것은 합성부에서 합성음을 합성하여 한 버퍼에 저장후 직렬 인터럽트에 의해 TDM 으로 데이터 전송을 하여 재생을 하고 있을 때, 다른 버퍼로 합성음을 저장하기 위해서이다. 한 쪽 버퍼의 음성이 모두 재생되었을 때, 데이터 포인터를 이동시켜 합성음 저장이 완료된 다른 버퍼를 재생하도록 하는 과정을 반복하면 실시간적인 처리를 할 수 있게 된다.

본 논문에서는 실시간 처리를 위하여 한 개의 재생 버퍼 크기를 2 초(16KB)에 해당하는 크기를 할당하여 사용하였다. 이에 따라 합성부에서는 재생 버퍼 크기만큼 합성 후 재생 버퍼에 데이터를 저장하고, 합성음

출력부에서 합성음이 2 초 동안 재생될 때 재생 버퍼 크기만큼 다시 합성하는 구조로 구성되었다.

3.3 다채널 확장

향상된 속도를 바탕으로 다채널 확장을 하기 위해 채널별로 보관해야 하는 정보와 데이터를 분석하여 채널별로 할당하였다. 채널별 정보와 데이터를 이용하여 다채널 TTS는 각 채널당 우선 순위를 정하지 않고 순차적으로 TTS 과정을 실시간적으로 수행한다. 본 논문에서는 그림 5와 같이 순차적인 TTS 과정을 수행할 때, 한 채널마다 최대 처리 시간 40 MHz 클록(=0.25 초) 이내에 2 초 이상의 합성음을 생성해 재생 버퍼로 데이터를 복사한 후 재생하도록 하여 8 채널의 실시간 TTS를 구현하였다. 8 채널의 확장으로 인해, 최초 채널에서 TTS 요구가 있은 후 마지막 채널이 응답하기까지 한 채널 당 최대 0.25 초의 처리 시간이 누적되어 최대 2 초의 지연이 수반된다.

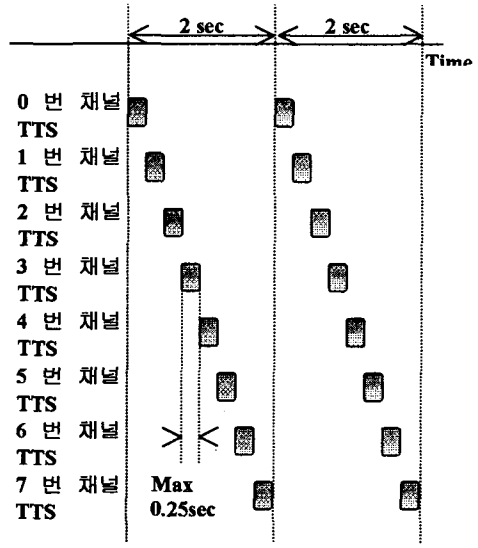

그림 5 다채널 TTS 타이밍도

IV. 시스템 실시간 테스트

본 논문에서 구현된 TTS 보드를 LG 정보통신에서 제작된 VIPS-HD II(Value-added Information Processing System – HD II[11])에 장착하여 테스트하였다. VIPS-HD II는 미리 저장된 음성을 사용자에게 들려주는 audiotex 서비스, 사서함을 통하여 가입자에게 음성메일, 팩스 메일을 주고 받는 메일 서비스를 기본으로 제공하고 그외 컴퓨터망을 연동시켜 호스트 컴퓨터의 데이터베이스에 있는 각종 정보를 음성 및 팩스로 제공해 다양한 음성 프로세싱 서비스를 제공할 수 있다.

다채널 실시간 테스트를 위하여, 시스템의 MPU-D(Main Processor Unit – Data processing module)를 TTS 보드의 각 채널로 TTS 명령과 텍스트를 동시에 송신할 수 있도록 프로그램을 구성하였다. 송신된 TTS 명령과 텍스트를 수신한 후 각 채널마다 실시간적으로 합성 및 재생을 하도록 하였고, 음성을 스피커에서 나오도록 장치하여 정상동작 여부를 확인하게 하였다. 테스트에 사용될 텍스트는 인터넷 게시판에 있는 16 KB 이내의 텍스트들을 무작위로 선택하였다.

MPU-D의 메시지 송신후 모든 채널의 스피커를 통해 음성을 청취하여 정상 동작됨을 확인하였다.

V. 결론

본 논문에서는 TTS 알고리듬을 16 비트 고정 소수점 DSP인 TMS320C6201을 이용해 다채널 실시간 구현하였으며, 실제로 음성 처리 부가 서비스 시스템에 보드 형태로 구현하여 응용하였다. 구현된 TTS는 최적화 작업을 통해 최대 40 MHz 클록으로 채널 당 2 초의 합성음 생성하도록 했으며, 개발된 TTS 보드는 두 개의 DSP를 사용하여 DSP 당 8 채널씩 총 16 채널을 수용하였다. 실험 결과, 모든 채널에서 실시간적으로 음성 합성이 수행됨을 확인하였다.

본 논문에서는 저가의 고정 소수점 DSP인 TMS320C6201을 사용했으나, 최근의 TTS의 알고리듬이 HMM과 같은 부동 소수점 연산을 많이 사용해가는 경향에 비추어 봤을 때, TMS320C67xx 같은 부동 소수점 DSP로 구현하는 것이 성능을 향상시킬 것으로 본다. 또한 대용량의 DB를 사용하는 TTS를 DSP로 구현할 경우 DB의 처리 및 사용에 좀 더 많은 연구를 한다면 성능을 향상시킬 것이라 판단한다.

참고문헌

[1] Thierry Dutoit, *An Introduction to Text-to-Speech Synthesis*, Kluwer Academic Publishers, 1997

[2] W.B Kleijn and K.K.Paliwal, *Speech Coding And Synthesis*, Elsevier, 1995.

[3] E.Harden, "High Quality Time Scale Modification of Speech Signals Using Fast Synchronized Overlap-Add Algorithm," *Proc.* ICASSP. pp. 409 – 412, 1990.

[4] 김세린, 이준우, 김상수, 이종석, 김민성, "한국어 문장-음성 변환 시스템에서의 운율 처리," 제 13 회 음성통신 및 신호처리 워크샵 논문집, pp. 415-418, 1996.

[5] 이준우, 김세린, 김상수, 이종석, 김민성, "수정된 음절을 이용한 한국어 문장-음성 변환 시스템," 제 13 회 음성통신 및 신호처리 워크샵 논문집, pp. 237-240, 1996.

[6] LG Corporate Institue of Technology, TTS 개발 완료 보고서, 1997.

[7] Texas Instruments, *TMS320C62x/67x Programmer's Guide*, 1999.

[8] Texas Instruments, *TMS320C62x/67x CPU and Instruction set*, 1999.

[9] Texas Instruments, *TMS320C6x Optimizing C Compiler*, 1998.

[10] Texas Instruments, *TMS320C62x/67x Peripherals*, 1999.

[11] LG information & Communications Ltd, *VIPS-HD II manual*, November, 1999.

[12] Rabiner, L.R. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc.* IEEE, Vol.77, No.2, 1989