포터블 기기의 실시간 처리를 위한 Job Scheduling에 관한 연구

장 석 우° , 박 인 규
홍익대학교 전기제어공학과
121-791 서울 마포구 상수동 72-1
penguni@netian.com

A study on realtime Job Scheduling for Portable Devices

Seock-woo Jang° , In-gyu Park
Dept, of Electrical & Control Eng. Honk Ik Univ.
72-1 Sangsu-dong Mapo-gu Seoul 121-791
penguni@netian.com

요약

Battery로 작동되고, 소형인 제품들도 다양한 기능은 물론이고, 다양한 입출력 장치를 갖추고, 실시간으로 처리하는 시스템이 많이 요구되고 있는 실정이며, 점차 더욱 더 요구될 것으로 전망된다. 더욱이 포터블 기기는 일반적으로 MCU의 내부에 제한된 ROM type 메모리를 내장하게 되면, 데이터 메모리로 SRAM 및 flash memory를 갖추고 있다. 따라서 이러한 제한된 하드웨어 환경하에서 많은 기능을 수행해야 하는 경우가 많다. 여러 기능을 시간적인 간격으로 배분하거나, 기능 자체를 서로 배분하면서, 서로 융합하는 등의 여러 가지 기능을 수행하려다 보면, 당연히 메인 소프트웨어 구조가 복잡해지며 대부분 어셈블리나 C와 같은 linear한 구조를 가지는 language로 개발되기 때문에 효과적인 프로그램 구조를 세우기는 쉽지 않다. 본 논문에서는 이를 위해 좀더 규격화된 방법을 제시하고자 한다.

보다 구체적인 구조를 연구할 목적으로 다양한 테스크를 수행하여야 하는 시스템이면서 프로세서가 필요한 포터블 기기의 한 응용 제품인 MP3 Player에서 요구되는 job scheduling을 연구한다.

필요한 작업의 종류는 가장 부하가 많이 걸리는 압축된 MP3 file을 다시 복원시켜주는 codec 부분과 일정 시간 간격을 가지고 수행하여야 하는 외부 키보드 입력과 실시간으로 시간을 계산하는 타이머 기능, 그리고 LCD에 시간의 변화를 표시하여 주어야 한다. 이와같이 수시로 작업이 발생하지만 시간 점유율이 중간 정도인 LCD 컨트롤과 메모리 컨트롤 등이다. 프로세서의 속도를 최소한으로 줄이면서 스케쥴링에 의해 시간 문제를 해결하는 방법을 제시하도록 한다.

I. 서론

기존의 작은 포터블 기기는 단순한 몇 가지의 작업만 하도록 설계되었다. 그러나 이제는 아무리 작은 시스템이라도 다양한 요구에 부응하기 위해 추가된 기능을 가지고 많은 작업을 수행해야만 한다. 이러한 작업중 프로세서 점유율이 높은 작업 또는 빈도가 잦은 작업등 다양한 특성을 가진 작업들이 있을 수 있다. 시스템 소프트웨어 개발자는 이러한 작업들이 최소한의 프로세서 사양을 충족하며 돌아갈 수 있도록 프로그래밍 하는데 노력을 한다.

여러 가지 작업들을 안정된 확신도를 가지고 수행되기 위해서는 고성능의 프로세서와 멀티 태스킹 OS의 사용이 가장 확실한 방법이다. 그러나 대부분의 포터블 시스템은 경제적인 면을 중요시 하는 경우가 많기 때문에 작은 시스템 사양이 바람직하다. 그리고 이러한 요구는 대부분 프로그래머의 기술과 노하우로 이루어지는 것이 일반적이다.

이 논문에서는 이러한 작은 프로그램의 요구를 좀더 체계적인 구조와 이러한 구조를 짜기전에 생각해야할 요소들을 제안하여 그 타당성을 보이고자 한다.

그리고 제안된 방법으로 구현한 프로그램을 시스

템에 적용시켜 그 결과를 확인한다. 프로세서 자체의 연산은 많이 필요로 하지 않으면서도 프로세서 점유율이 일정하게 높은 데이터 전송 작업을 대신하는 CODEC control과 수시로 이루어져야 하는 Key Interface ,그리고 때때로 일어나는 중간 정도의 프로세서 점유율을 가진 LCD control 등으로 구성 되어 있는 MP3 player system에 대하여 설명한다.

Ⅱ. 적용 시스템

본 논문에서 논의 될 시스템의 기능은 그림 1에서 보듯이 다음과 같은 기능을 한다.

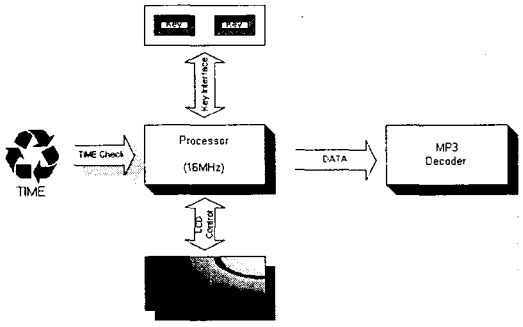

그림 1 적용 시스템

첫째 MP3 디코딩시 디코더에 데이터 스트림을 보내주게 된다. 이는 연속적으로 일어나야 하며 프로세서의 부하를 가장 많이 먹히게하는 작업이다.

둘째 사용자의 키 입력에 즉각 반응하기 위해 포트 입출력 단자를 매트릭스 모습으로 이루어진 Key 인터페이스를 일정 간격을 두고 체크해야 한다. 이는 사용자가 딜레이를 느끼지 못할 만큼의 시간 간격을 두고 작업이 이루어져야 한다.

셋째는 수시로 일어나는 LCD 평판 디스플레이 표시를 실시간으로 부드럽게 하기 위한 LCD control 작업을 들수 있다. 특별히 노래의 흘름에 따른 시간의 흐름 표시, 노래 가사의 표시, 노래 제목, 가수명 등이 하나의 화면상에 동시에 표시하여 줄 수 없기 때문에, 일부씩 흐른 모습으로 표시하여야 주어야 한다.

넷째 전체 작업을 위한 시간 흐름을 체크하고, 동시에 배터리의 용량 상태를 체크하면서, 배터리 용량이 일정 이하로 내려 갈 경우에는 사용자에게 경고 메시지를 주어야 한다.

재생될 MP3 파일은 최대 스트림인 128kbps로 제한한다.

Ⅲ. 프로그램 구조

프로그램 구조를 평가하는 파라미터에 대한 정의를 먼저 하기로 한다.

작업	프로세서 점유율	발생 빈도
Decoder control	70%	특정 상태 하에서 항상
Key Interface	5% 미만	0.3초 이내 마다
LCD control	10%	특정 상태로 변화시
Time Check	5%미만	매 단위시간(초)마다

표 1 작업 특성

어떤 작업에 따른 프로세서 점유율이란 해당 작동 수행을 위하여 프로세서가 작동하여야 하는 기간으로 정의한다. 다른 작업과 함께 수행할 때 점유하는 시간을 실제로 테스트 해보기 전에 알수는 없으므로 윗 시간은 모델링시 임의로 정하는 추측값인 것이다. 또 다른 파라미터로 발생 빈도는 특정 작동이 전체 시스템에서 얼마나 자주 작동되어야 하는가 하는 정도를 나타낸다.

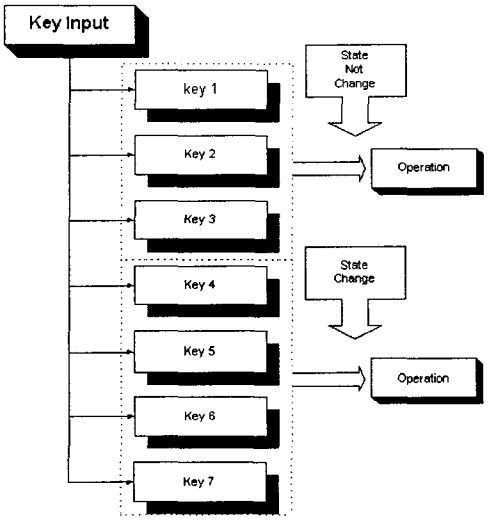

그림 2 Key Control Flow

물론 이러한 평가 기준 파라미터는 현재 작동되고 있는 작업에 따라, 또한 특정 작업 중에서 특정 모드에 따라 당연히 다르게 된다. 작업 모드는 주로 키 입력에 따라 달라지므로 모드는 그림 2에서 보듯 키 입력 중심으로 전환되어야 한다. 그림에서 처럼 키 스캔은 계속 포트 값을 읽어서 포트의 값이 변했는

가 변하지 않았는가를 체크하여 해당 비트의 버튼이 눌렸는가 떨어졌는가를 알아낸다. 이 작업으로 키 입력을 받으면 현재 상태가 무엇이었는가에 따라 다음의 정해진 상태로 넘어가게 된다. 곧 상태가 변하기 위해서 항상 키 스캔이 일어나야 한다.

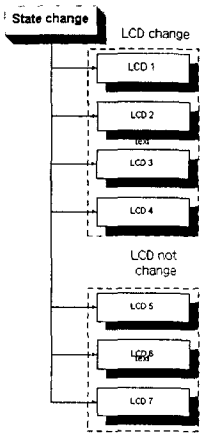

그림 3 LCD control Flow

LCD의 디스플레이는 그림 3와 같이 모드가 바뀌는 때마다 다른 그림, 폰트로서 해당 모습이 나타나야 하고, 같은 모드 안에서도, 실시간으로 시간의 흐름의 모습으로 숫자가 초 단위로 계속 변하여야 한다. 또한 가수 면 혹은 노래 제목 등이 화면 하단에 왼쪽부터 오른쪽으로 향하면서 글자가 계속 흐르는 모습으로 표시하여 주어야 한다. LCD의 출력도 상태가 변함에 따라 이루어지며 그러므로 LCD 컨트롤 루틴은 상태가 변한 후에 일어나야 한다.

이러한 LCD의 화면이 변하는 상태에서도 키 입력이 제대로 작동하여 주어야 하므로 MP3 플레이시 이 모든 작업이 같이 일어나는 플로우는 그림4와 같다. 이 그림은 플레이 모드의 플로우이다. 원래 이 작업뿐 아니라 배터리 체크나 다른 외부 입력등의 인터럽트도 체크해야 하는 경우가 많이 있다.

이 경우를 보면 플레이로 모드가 전환되었을 경우 LCD 출력이 일어나고 그 후 해당 키가 눌려 모드가 바뀔 때 까지 키 스캔을 반복하는 구조로 되어 있다. 이러한 위에서 언급한 여러 요구조건을 만족하는 구조로 그림 5와 같이 메인 플로우를 구성하게 된다.

압축된 MP3 노래 file이 flash 메모리에 저장되어 있어, 이러한 flash 메모리로부터 MP3 Decoder로의 데이터 전송은 MP3를 플레이 하는 특정 모드에서 작업이 수행되고 또한 프로세서의 70%이상을 점

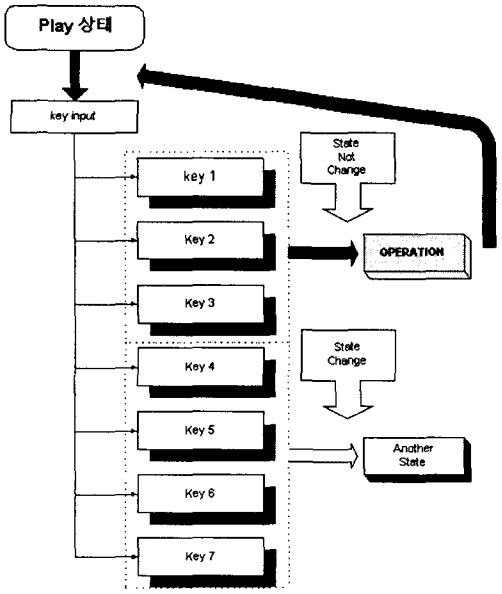

그림 4 Play State

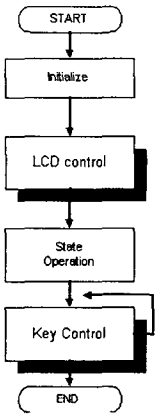

그림 5 Main Flow

유하는 작업이므로 항상 수행되어야 하며 다른 작업에 긴 시간 방해 받아서는 안된다. 그러므로 정의된 특정 상태내에서는 이 작업이 연속적으로 일어나야 한다.여기에 별도로 시간을 체크하는 루틴이 들어가야 한다. 이는 플레이시 사용자에게 보여지는 결과이기도 하고 MP3 파일의 정보를 업데이트 시키기 위한 작업이기도 하다. 이 경우는 메인 플로우와는 상관없이 타이머 인터럽트를 사용하여 구성한다. 그러나 앞서 밝힌바와 마찬가지로 프로세서 점유율이 높은 플레이등의 작업을 방해 하면 안되므로 시간

체크와 병행되어야 할 LCD 디스플레이도 인터럽트 서비스 루틴에 포함시키기는 어렵다. 그러므로 인터럽트 서비스 루틴에서는 시간만을 체크하고 플레이시 반복되는 루프의 적당한 위치에 갱신된 시간을 디스플레이 하는 루틴을 넣어야 한다. 이는 그림 6과 같이 표현된다.

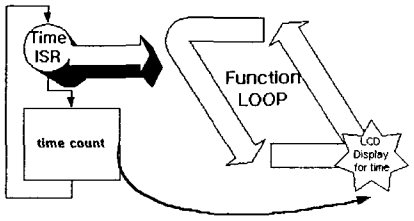

그림 6 Time ISR

모든 작업을 포함한 구조를 만들어 보면 그림 7과 같다.

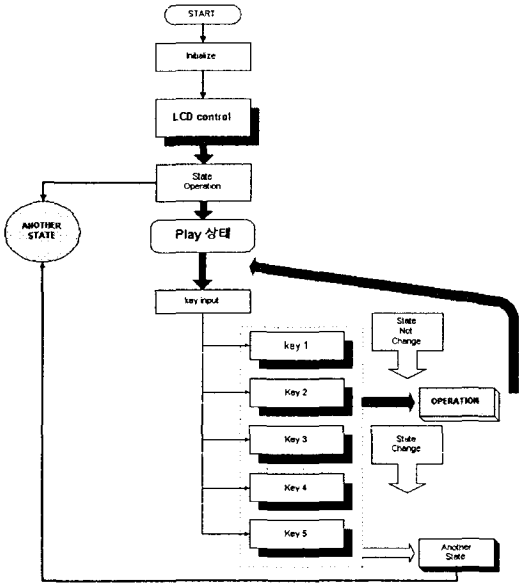

그림 7 Total Flow

Ⅳ. 테스트 환경

MP3 player를 구성하는 테스트 환경은 프로세서 칩으로 PIC17C756 RISC 8-bit processor이고, MP3 decoder로서 MAS3507 MPEG AUDIO Decoder, 칩이 사용되고, 64x 132 bit-map LCD 모듈과 데이터가 저장되어 있는 6개의 16K x 8 Flash Memory 및 MAS3550 D/A converter 및 USB controller 칩으로 구성되어 있다.

Decoder의 데이터 흐름은 시리얼 버스를 통해 일어나며 LCD controller의 데이터 흐름은 패러랠 버스를 통해 일어난다. Test bitstream은 128Kbps MPEG1 Layer3 스트림을 사용하였고 기능들은 실제 MP3 player 개발 시나리오 기능을 구현했다. 프로세서 속도는 33Mhz에서 8MHz를 임의로 적용하며 테스트 하였다.

Ⅴ. 결과

테스트 프로그램에서 프로세서 수행속도를 결정시켜주는 가장 큰 요소는 부하를 가장 많이 차지하는 Decoder control이다. 128kbps 속도의 스트림을 디코딩 하기 위해서는 실제적인 통신 프로그램을을 제외하고라도 128k*2(클럭과 데이터) 만큼의 포트 접근용 프로그램을 필요로 하고 메모리에서 읽어오는 데에도 총 길이의 1/8만큼이 소요된다. 이러한 조건에서도 이론적으로는 8MHz 프로세서 클럭만으로도 동작해야만 하지만 실제로 압축화일을 복원시켜 플레이만 하는데에도 8MHz로는 불가능하였다.

16MHz에의 프로세서 속도에서 70%정도의 프로세서 점유율로 나타났고, 이러한 16MHz 클럭을 사용하였을 때 앞에서 제안된 구조에 잘 맞게 프로그램이 동작하였다.

위와 같은 프로그램 구조는 멀티 태스킹의 기본 구조인 메시지 큐의 형태를 따라간다. 그러면서도 한 가지 작업에 집중적일 수 있는 선형성을 가지게 되므로 프로세서 점유율의 크리티컬 레인지를 잘 맞춰 갈 수 있을 것이다. 또 그러한 구조를 사용하므로써 추가된 작업들이 있는 경우에도, 쉽게 확장하고 타이밍을 맞춰 시스템 성능을 향상 시킬 수 있을 것이다.

참고문헌

[1] 멀티 프로세서를 위한 타스크 스케쥴링 기법 1992

[2] 다중스레드 컴퓨터를 위한 효율적 스케쥴링 기법 1994

[3] 프로세스 분담처리에 의한 효율적인 루프 스케쥴링 방법, 1992