MicroC/OS-II 운영체제에서의 순간 최대전력 분산 기법

우장복° 서효중
가톨릭대학교 컴퓨터공학과
{sofe4u°, hjsuh}@catholic.ac.kr

Peak Power Distribution for MicroC/OS-II

Jangbok Woo°, Hyo-Joong Suh
Dept of Computer Science and Engineering, The Catholic University of Korea

요 약

최근 PDA, PMP, 핸드폰 등 휴대용 임베디드 기기의 사용이 증가되고 기능이 점점 다양해지며, 고성능을 추구하게 됨으로써 전력 소모 역시 증가하게 되었다. 휴대용 임베디드 기기는 대부분 배터리를 기반으로 동작하므로 에너지원이 제한적이어서 한정된 에너지원을 효율적으로 사용하는 전력관리 기법에 대한 연구가 많은 관심을 받고 있다. 시스템 사용시간의 연장을 위해 시스템의 성능 저하를 최소화하면서 소모되는 전력을 최소화하기 위한 여러 방법 들이 제시되었으나, 기존의 방법들은 각각의 방전 패턴에 따라서 사용시간이 달라지는 배터리의 특성을 고려하지 않고 주로 시스템의 평균 전력 소비 감소만을 목적으로 한다. 이에 본 논문에서는 배터리의 방전 특성을 고려하여 휴대용 임베디드 기기에서 배터리의 사용시간을 연장할 수 있도록 MicroC/OS-II 운영체제에서의 순간 최대전력 분산 기법을 제안한다.

1. 서 론

최근 PDA, 핸드폰 등의 무선통신 단말기 외에도 PMP(Portable Multimedia Player), 휴대용 게임기, 노트북 컴퓨터 등 배터리를 에너지원으로 동작하는 휴대용 임베디드 기기가 사회 전반에 걸쳐서 폭넓게 사용됨에 따라 배터리의 한정된 에너지원을 효율적으로 사용하는 전력관리 기법에 대한 연구가 활발하게 이루어지고 있다. 전력관리란 시스템의 성능 저하를 최소화하면서 소모되는 전력을 최소화하기 위해 전력을 소모하는 기기를 관리하는 것을 말한다. 효율적인 전력관리를 위해서는 시스템 내의 전력 기기들 중에서 전력소모가 큰 기기에 대한 정보가 필요하고 단위 시간당 수행된 작업의 양을 최대화해야 하며 배터리의 용량 및 다른 기기들에 대한 영향을 고려하여야 한다[1].

전력관리 기법에는 기본적으로 시스템이나 기기가 사용되지 않을 때는 해당 기기를 저전력 상태로 동작시켜서 전력소모의 낭비를 막는 동적 전력관리 기법(Dynamic Power Management)[3]과 프로세서의 전력소모가 동작 주파수에 비례하고 또한 구동전압의 제곱에 비례하는 점을 고려하여, 프로세서의 동작 주파수 및 구동전압의 변동을 통해서 전력 소모를 감소시키는 동적 전압조절 기법(Dynamic Voltage Scaling)이 있다[4]. 이러한 전력관리 기법들은 주로 전체 시스템이나 시스템 내의 전력 기기들의 평균 전력 소비 감소를 목적으로 하여 시스템을 구성하는 기기들의 전력소모 양을 최소화하여 주는 장점이 있다. 그러나 사용패턴 및 프로파일에 따라서 소모량이 달라지는 배터리의 특성을 고려하지 않았기 때문에, 일정 시간 동안 시스템의 사용이 집중적으로 일어나는 경우 순간 최대전력이 지속적으로 발생하여 배터리의 사용 효율성이 저하되는 단점이 있다[2]. 본 논문에서는 배터리의 방전 특성을 고려하여 휴대용 임베디드 기기에서 배터리의 사용시간을 연장할 수 있도록 MicroC/OS-II 운영체제에서 일정 시간구간 동안 사용가능한 에너지의 양을 정해놓고, 시스템의 에너지 소모가 해당 기준을 초과할 경우 현재 수행되는 작업을 임의의 클럭 틱(Clock Tick) 동안 지연 실행시켜서 순간 최대전력을 분산시키는 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구에 대해서 기술하고, 3장에서는 제안하는 기법에 대해서 설명하고, 4장에서는 제안하는 기법의 성능을 평가하며, 5장에서는 결론 및 향후 연구과제에 대해서 기술한다.

2. 관련연구

2.1 MicroC/OS-II

MicroC/OS-II는 1992년에 Jean J. Labrosse에 의해 개발된 선점형 실시간 멀티태스킹 임베디드 운영체제이다. MicroC/OS-II는 비상업적인 용도에 한해서 소스코드를 자유롭게 사용할 수 있으며, 대부분의 8, 16, 32, 64 비트의 프로세서들과 DSP(Digital Signal Processor)로도 포팅할 수 있고, 임베디드 제품에 내장할 수 있는 내장성과 필요한 기능만 이미지에 포함할 수 있는 유연성을 갖는다. 또한 선점형 실시간 멀티태스킹을 지원하고 함수의 실행시간이 확정적이며 각 태스크의 스택 크기를 다르게 설정할 수 있는 특징이 있으며, 안전 결정적인 시스템이 필요로 하는 표준인 RTCA DO-178B 규격을 만족하여 현재 카메라에서 항공장비에 이르기까지 다양한 제품에 사용되고 있다[5].

2.2 MicroC/OS-II의 태스크

태스크(Task)란 일정기간 동안 프로세서의 자원을 차지하는 간단한 프로그램을 의미하며 쓰레드(Thread)라고도 불린다. MicroC/OS-II는 64개의 태스크를 지원하며 선점형 실시간 커널을 채택하여 항상 높은 우선순위의 태스크가 프로세서를 점유하게 된다. 이 때, 태스크의 우선순위는 응용 프로그램이 실행되는 동안에도 지속적으로 변하는 동적 우선순위이다[5]. MicroC/OS-II의 태스크는 DORMANT(수면상태), READY(준비상태), RUNNING(실행상태), WAITING(대기상태), ISR(Interrupt Service Routine)의 총 5가지 상태를 가지며, 아래 그림 1과 같이 프로그램 실행에 따라 태스크의 상태 변화가 일어난다.

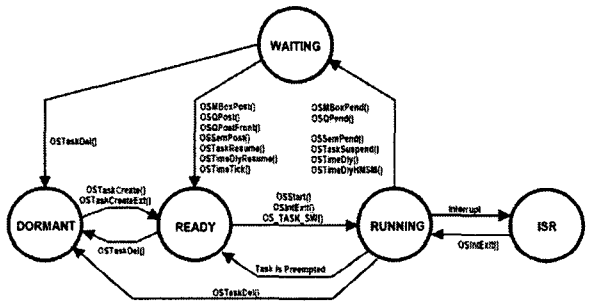

그림 1 태스크의 상태 변화도

2.3 배터리의 특성

휴대용 임베디드 기기에 사용되는 배터리는 소형, 경량화 및 장시간 연속사용 등의 요구를 만족하기 위해서 보통 리튬이온(Li-ion) 등의 2차 전지가 사용된다. 배터리는 충전을 통해서 전기 에너지가 화학 에너지로 변환되어 저장되며, 방전을 통해서 화학 에너지가 전기 에너지로 변환되어 기기에 전력을 공급하게 된다. 이 때 배터리는 사용패턴 및 프로파일에 따라 방전시 에너지의 소모량이 달라진다.

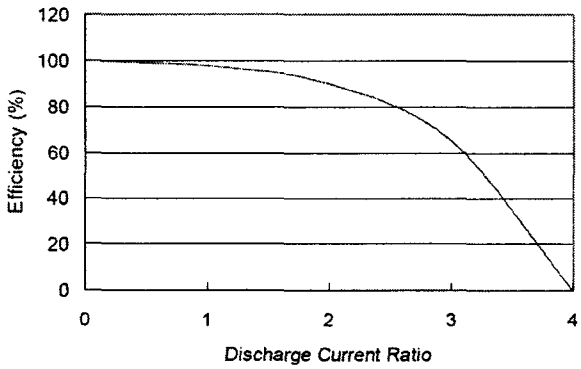

그림 2 배터리의 방전 특성

위의 그림 2에서 알 수 있듯이 같은 시간을 사용하더라도 연속적으로 사용하는 것이 드문드문 사용하는 것보다 많은 에너지를 소모하며, 큰 전류가 흐를 경우 배터리 전압이 차단 전압으로 일찍 감소하여 발생 가능한 전하의 양이 줄어들어 제공할 수 있는 전력량이 감소하므로 방전전류의 증가에 따라 배터리의 사용 효율성이 비선형적으로 감소하는 특성을 보인다[6].

3. 제안하는 기법

MicroC/OS-II는 태스크가 생성된 직후 스케줄링이 시작되며, 태스크의 상태를 관리하기 위한 커널 내부 자료구조인 태스크 컨트롤 블록(Task Control Block)을 포함하고 있다. 태스크 컨트롤 블록은 "OS_CORE.c"에 정의되어 있고 태스크가 생성될 때 정해진 내용으로 초기화된다. 본 논문에서는 시스템에서 발생하는 순간 최대전력을 분산시키기 위해서 아래 그림 3과 같이 변수를 추가적으로 선언하였다.

```
typedef struct os_tcb {
        .....
// peak power distribution을 위한 변수 선언
   INT16U     peak_th          // 기준 값
   INT16U     power_tot        // 소모 전력의 합
        .....
} OS_TCB;
```

그림 3 OS_TCB의 변수 추가 부분

태스크가 소모하는 전력은 입출력 디바이스의 디바이스 드라이버의 수정을 통해서 구할 수 있으며, 데이터 입·출력이 발생할 때 소모되는 전력과 수행되는 데이터의 양의 곱으로 소비할 에너지를 구할 수 있다. 프로그램이 실행 되서 프로세서가 작업을 수행하면 power_tot의 값이 증가하게 되고, 프로세서가 유휴상태가 되면 값이 감소하게 된다. power_tot의 값이 계속 증가하여 순간 최대전력을 분산시키기 위한 기준 값인 peak_th 값을 초과하면, 실행 중이던 태스크는 대기상태가 되고 OSTimeTick() 함수가 호출되어 클럭 틱 인터럽트가 발생된다.

클럭 틱은 정기적으로 일어나는 특별한 인터럽트이며, MicroC/OS-II에서 인터럽트 사이의 시간은 응용프로그램에 따라 달라지지만 일반적으로 10ms에서 200ms 정도이다[5]. 클럭 틱 인터럽트는 커널로 하여금 클럭 틱 주기의 정수배만큼 태스크를 지연시키고, 이벤트가 발생하기를 기다리는 태스크에게 타임아웃을 제공한다. 클럭 틱 인터럽트는 power_tot의 값이 peak_th 값보다 작아질 때 까지 계속 수행되며, power_tot의 값이 peak_th 값보다 작아지게 되면 대기 중이던 태스크 중에서 우선순위가 가장 높은 태스크가 실행된다. 그림 4는 제안하는 기법의 수행과정을 나타낸다.

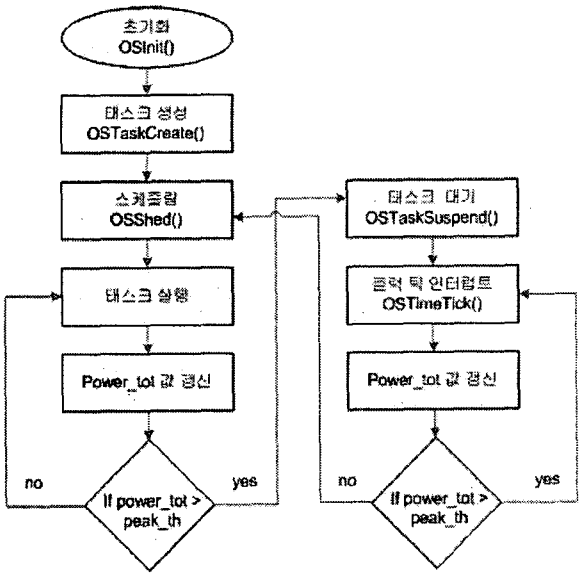

그림 4 제안하는 기법의 수행과정

4. 성능평가

본 절에서는 3절에서 제시하는 기법의 성능을 검증하기 위해서 기존의 MicroC/OS-II를 적용했을 때와 제안하는 기법을 적용했을 때의 배터리 사용 효율성을 비교하였다.

4.1 실험 환경

리튬이온 2차 전지를 배터리로 사용하는 PMP에서 하드디스크에 저장된 영상을 재생할 때, 주기적으로 전력을 소모하고 순간 최대전력 소모를 발생하는 태스크에는 프로세서를 통한 영상 디코딩, LCD를 통한 화면 재생, 하드디스크를 통한 읽기 수행이 있다고 가정하였다. 또한, 그 외의 디바이스들은 전체 시스템의 배터리 사용 효율성에 영향을 미치지 않으며, 데이터의 크기에 따라서 입출력에 소모되는 시간은 비례한다고 가정하였다.

그리고 PMP에서 하드 디스크에 저장되어 있는 영상이 2시간 동안 재생될 때, 순간 최대전력의 발생 비율이 전체 재생시간의 1%, 5%, 10%, 20%일 때의 배터리 사용 효율성을 각각 비교하였다.

4.2 실험 결과

그림 5는 순간 최대전력의 발생 비율을 변화시키며 실험했을 때의 배터리 사용 효율성을 나타낸다. 순간 최대전력이 발생하는 시간 구간에서 제안하는 기법을 적용했을 때가 기존의 MicroC/OS-II를 그대로 적용했을 때 보다 평균적으로 1.28배 정도의 배터리 사용 효율성을 보이는 것으로 나타났다.

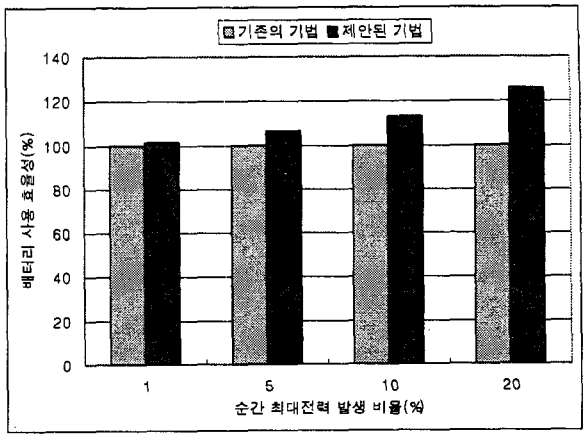

그림 5 배터리 사용 효율성 비교

5. 결론 및 향후과제

본 논문에서는 MicroC/OS-II 운영체제에서 일정 시간 구간 동안 사용가능한 에너지의 양을 정해놓고, 태스크들이 수행될 때 발생하는 에너지 소모가 해당 기준을 초과하여 순간 최대전력이 발생하는 경우 인터럽트를 발생시켜서 현재 수행되는 작업을 임의의 클럭 틱 동안 지연 실행시킴으로써 순간 최대전력을 분산시켜서 결과적으로 배터리의 사용시간을 연장시킬 수 있는 기법을 제안하였다. 향후 연과과제는 제안된 기법을 휴대용 임베디드 기기에 실제로 적용하여 결과 값을 측정해보고, 시스템의 성능저하를 최소화하면서 배터리의 사용시간을 연장하는 효과적인 순간 최대전력 분산 기법을 연구하는 것이다.

참고문헌

[1] D. Ramanathan, S. Irani, R. Gupta, "An Analysis of System Level Power Management Algorithms and Their Effects on Latency", IEEE Trans. on Computer Aided Design of Integrated Circuits and System, vol. 21, no. 3, Mar. 2002.

[2] Thomas L. Martin, et al., "A Case Study of a System-Level Approach to Power-Aware Computing", ACM Transactions on Embedded Computing Systems, August, 2003

[3] T. Simunic, L. Benini, P. Glynn, G. De Micheli, "Dynamic Power Management for Portable Systems", in Proceeding. the sixth Annual International Conference on Mobile Computing and Networking, pp. 11-19, 2000.

[4] P. Pillai, Kang G. Shin, "Real time Dynamic voltage scaling for low power embedded operating systems", in Proceedings of the 18th ACM symposium on Operating System Principles, pp. 89-102, 2001

[5] Jean J. Labrosse, "MicroC/OS-II The Real-Time Kernel", R & D Technical Books, 1998

[6] D. Linden, T. Reddy, "Handbook of Batteries", McGraw-Hill, NY. 2001.