

로봇 제어용 리눅스 기반 실시간 커널의 설계 및 구현

Design and Implementation of Linux based Real-Time Kernel for Robot Control

노현창*, 고낙용**, 김태영***

* 조선대학교 제어계측공학과(Tel : 82-062-230-7108; Fax : 82-062-224-1987 ; E-mail : arcs@shinbiro.com)
** 조선대학교 제어계측공학과(Tel : 82-062-230-7108; Fax : 82-062-224-1987 ; E-mail: nyko@mail.chosun.ac.kr)
*** 조선대학교 제어계측공학과(Tel : 82-062-230-7108 ; E-mail : robot@stmail.chosun.ac.kr)

Abstract: This paper presents a method for building a real-time kernel of autonomous mobile robot control systems. Until now, most of robots have their own operation softwares dedicated only for their use. Sometimes, operation softwares were developed based on MS-DOS or other real-time kernel based on UNIX. However, MS-DOS has many restrictions for use as a robot operation system. Also, UNIX based real-time kernel has some limitations for use with mobile robots. So, in this paper, we focus on building a real-time kernel based on Linux. The in this paper, the software modules of Task Management, Memory Management, Intertask Communication, and Synchronization are redesigned. To show the efficiency of the paper, it was applied to run Nomad Super Scout II avoiding obstacles detected by sonar sensor array.

Keywords : Real-Time, Kernel, RTOS, Nomad Scout II, Linux

1. 서론

현재까지는 대부분 이동로봇에 필요한 제어기를 직접 제작을 하거나, DOS 기반 하에 연구가 되어왔다[1]. 그러나 이러한 방법은 시시각각으로 변화하는 모든 주변 환경을 적절하게 대응해 나갈 수 없다는 단점이 있고, 여러 개의 하드웨어 인터럽트를 빈번하게 이용해야 했다. 이것은 로봇 제어를 위한 응용 프로그램을 작성하는 작업을 힘들게 했으며, 실시간 시스템의 특징을 갖고 있는 로봇 제어기의 기능을 여러 면에서 제약하는 요인이 되었다. 따라서 본 논문에서는 현재 관심이 집중되어 지고 있는 Linux 기반 하에 실시간 커널을 설계한다. 일반적으로 실시간 커널은 운영체제의 커널 모드에서 제작한다. 그러나 이러한 방법은 실시간 시스템으로서의 성능은 좋을 수 있으나, 특정 시스템에 종속되어 진다. 따라서 본 연구에서는 시스템의 커널 모드가 아닌 사용자 모드에서 제작을 한다. 본 연구에서 개발한 실시간 커널은 POSIX/ANSI-C 기반 하에 설계하였고, 실시간 커널에 필요한 구성요소인 스케줄러, 태스크 관리, 태스크간 통신, 태스크간 동기화 같은 구성 요소를 재 설계하였다[2,3].

2. 실시간 시스템

실시간 시스템이란 어떤 이벤트가 발생했을 때 이것이 어떤 정해진 시간 이내에 처리되는 것을 보장하는 시스템이다. 즉 이벤트에 대한 반응이 빠르고, 중요한 이벤트가 덜 중요한 이벤트보다 먼저 수행되며, 이벤트를 놓치는 일이 결코 일어나서는 안 되는 시스템을 말한다. 실시간 커널은 태스크의 시간제약 조건을 고려하고 실행시간의 부담을 줄이며 시스템의 전체적인 수행을 빠르게 하기 위하여 빠른 문맥 교환, 최소의 기능만을 보유한 작은 크기의 커널, 외부 인터럽트의 빠른 반응, 우선 순위에 기반을 둔 스케줄링, 실시간 시계(Real-Time Clock), 프로세스의 상호 배제, 태스크간의 통신(Intertask

Communication)의 기능들을 제공하면서 구성되어진다.

일반적으로 말하는 RTOS는 실제적으로는 존재하지 않는다. 즉 OS에다가 실시간 기능을 추가해서 RTOS라 부르는 것이다. 실시간 커널의 구조를 살펴보면 다음과 같다.

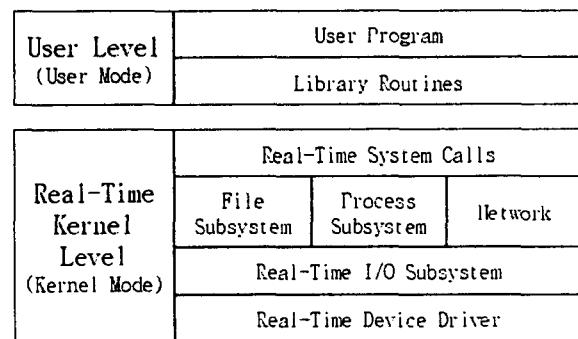


그림 1. Real-Time Kernel Structure

그림 1은 운영체제의 부분을 제외한 일반적인 실시간 커널의 계층 구조이다. 실시간 커널 부분은 운영체제의 커널 부분에 포함이 되거나 혹은 독립적으로 존재하게 된다. 그림 1에서 보았듯이 실시간 커널의 구조는 운영체제 커널의 구조와 별 차이점이 없다. 그러나 커널과 인터페이스를 하는 함수 부분과 하드웨어를 접근하는 장치 드라이버 부분은 일반적인 커널과는 달리, 이러한 부분도 실시간 요소가 포함이 된다.

리눅스가 Real-Time 기능을 갖추려면 실제로 커널부터 다시 설계되어야 하지만, 이는 커널을 구성하는데 많은 시간을 필요로 하며, 리눅스 커널이 업그레이드 될 때마다 이를 신속히 반영하기 어렵고, 버그가 생길 가능성이 더 커지게 된다. 이에 RT-Linux는 커널을 다시 설계하지 않고 리눅스 소스