제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

리얼 타임 리눅스 기반 개방형 로봇 제어기

신주호, 문승빈
세종대학교 컴퓨터공학과
e-mail: yabarahi@robotics.sejong.ac.kr

The Real-Time Linux Based Robot Controller

Jooho Shin, Seungbin Moon
Dept. of Computer Engineering, Sejong University

요 약

본 논문은 Real-Time Linux를 기반으로 하는 개방형 로봇 제어기에 관한 내용이다. 여기서 Real-Time OS는 시스템의 수행결과가 기능적으로 정확해야 할뿐만 아니라, 결과가 도출되는 시간 역시 주어진 제약 조건을 만족시켜야 하는 시스템이 이다. 이러한 특징 때문에 RTOS(Real-Time OS)는 항공, 전자, 기계분야뿐만 아니라 네트워크 분야에서도 많이 적용되어지고 있다. 본 논문에서는 Open Source를 지향하고 있는 범용의 Linux를 기반으로 하는 Real-Time Linux를 이용하여 로봇을 제어하는 구현을 함으로서 Real-Time Linux의 제어분야에 적용가능성을 제시해 본다.

1.서론

이 논문은 Real-Time Linux를 이용한 로봇제어기 구현에 관한 글이다. 현재 로봇제어기를 구현할 때 RTOS를 많이 이용하고 있다. 그 이유는 로봇 제어기는 시스템의 수행결과가 기능적으로 정확해야 할뿐만 아니라, 결과가 도출되는 시간 역시 주어진 제약 조건을 만족시켜야 하는 시스템이기 때문이다 이러한 조건을 모두 만족 시키기 위해 RTOS를 사용한다. 하지만 기존의 상용 RTOS들은 제품 가격도 고가이고 경우에 따라 ROM-Copy License등을 포함시키는 경우가 많아 대기업이 아닌 일반 중소기업등이 사용하기에는 많은 어려움이 있었다. 이에 반해 Linux는 소스가 공개 되어 있고 소스에 관한 사용료가 없다. 뿐만 아니라 여러 공동체에서 Linux를 이용한 다양한 응용에 관한 연구를 진행하고 있고 그 결과를 공개 하고 있다. 때문에 대학 연구소 뿐만 아니라 현재에는 많은 기업에서 적용시키고 있다. 여기서 Real Time Linux라는 것은 범용의 Linux kernel에 Real-Time의 조건을 만족시킬수 있는 기능들을 더한 Linux로 현재 항공, 제어, 통신 분야에서 많은 연구가 이루어 지고 있다. 이에 따라 본 논문을 통해 Real Time Linux를 이용한 로봇제어기를 직접 구현해 봄으로서 제어분야에서의 Linux 사용에 대한 가능성을 보이고자 한다. 본 논문의 구성은 사용된 RTOS의 소개와 제어기의 구성

2.RTOS(Real-Time OS)

본논문에서 사용된 RTOS는 DIAPM(Dipartimento di Ingengneria Aerospaziale, Politecnico di Milano)에서 만든 RTAI(Real Time Application Interface, for Linux) Kernel로서 x86계열의 PC 기반 RTOS이다. 지금은 x86시스템 뿐만 아니라 ARM, PPC계열의 CPU도 지원하고 있다[1][2][3].

3. 제어기의 구성

제어기는 소프트웨어와 하드웨어 부분으로 나누어진다. 그리고 소프트웨어 부분에서 다시 User Space과 Kernel Space 공간으로 다시 나누어 진다. 이 부

분은 아래 그림 1과 그림 2에 나타난다.

3.1 Kernel Space Tasks

그림 1은 Kernel Space에서의 하는 일을 나타낸다. 정확한 Real-Time 특성을 보장 받기위해서는 Kernel Space에서 프로그래밍을 해야 한다. 따라서 Time-Critical한 작업을 요청하는 부분은 Kernel 내에서 프로그래밍하였다. 그 작업 내용들은 로봇의 모션을 실시간으로 처리하는 Task, I/O Control, Servo Driver의 현재상태를 모니터링 하는 Task들로 구성되어 작업되어진다.

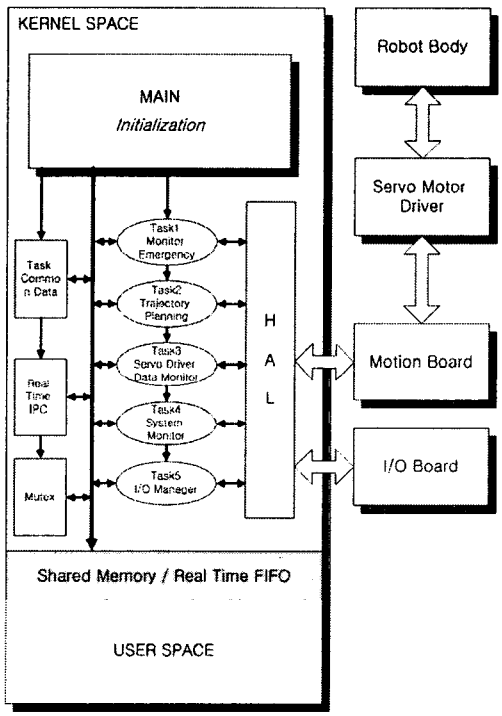

그림 1 Kernel Space Tasks

3.2 User Space Tasks

그림 2는 User Space 에서 이루어지는 작업들을 나타낸다. User Space에는 상대적으로 Time-Critical한 기능이 필요 없는 Task들로 이루어진다. 구성은 사용자 인터페이스 부분인 GUI와 RS-232C로 로봇을 지시를 받을수 있는 Protocol 처리 부분, Robot Language 처리부분등이다. 각각의 Task를 처리하기 위해 기존의 범용 Linux가 제공하는 POSIX Library를 사용하지 않고 RTAI에서 제공하는 LXRT Library[4]를 사용하였다. 이 Library는 Hard Real-Time 특성을 보장 받지는 못하지만 Soft Real-Time 특성을 보장할수 있는 POSIX Library이다. 여기서 지시받은 내용들은 각각 조건을 체크 받고 Real-Time의 특성을 보장 받는 Shared Memory와 FIFO를 통해 명령이나 데이터를 Kernel Space로 보내게 된다. GUI에서 사용되어 지는 Graphic Library의 제한을 없애기 위해 중간에 Shared Memory를 통해 로봇을 제어할수 있게 하였다. 따라서 어떠한 Graphic Library를 사용하건 Shared Memory를 통한 프로토콜 규약을 준수하면 된다. 본 논문에서는 QT[5]를 사용하였다.

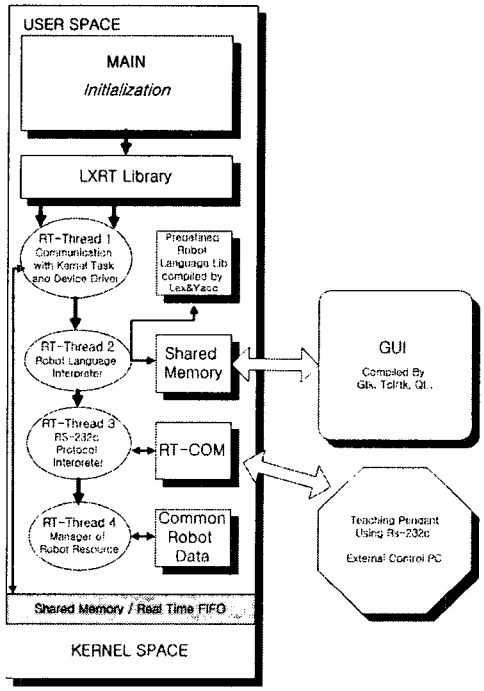

그림 2 User Space Tasks

4. 로봇 언어

로봇 언어는 사용자와 로봇간에 사용되는 대화 수단으로 교시된 위치 또는 변수 조작을 통해 로봇을 동작시키는 언어를 말한다. 로봇 언어의 사용은 시스템의 구조적인 변경 없이 단지 사용자가 로봇 언어를 이용하여 작성된 프로그램을 실행 시켜 다양한 작업을 구현할 수 있게 한다. 로봇 언어가 필요한 것

은 로봇이 단순한 작업을 벗어나 보다 복잡하고 다양한 작업을 구현하기 위해서 필요하다. 개방형 로봇 언어가 필요한 것은 많은 로봇 언어들은 이식성을 고려하지 않게 만들어지기 때문에 만약 다른 로봇 언어를 작성한다면 새로운 작업을 해야 한다. 이에 개방형 로봇 언어는 이식성을 고려한 다양한 로봇 언어를 구사하여 이식성을 높이기 위해 제작되었다. 로봇 언어 Interpreter는 Lex & Yacc[6]로 Compile 되었다.

5. 제어기의 하드웨어

제어기의 하드웨어는 크게 3부분으로 구성된다. 상위제어기 역할을 하는 산업PC가 있고, PC로부터 연산된 결과를 모터에 전달하는 Motion Board, 마지막 로봇이다.

5.1 상위제어기

그림 3은 상위제어기로 사용되어진 산업용PC의 모습이다. 사양은 Pentium-500Mhz, ram 256MB 이다.

그림 3 상위제어기의 모습

5.2 모션보드

모션보드(DSP Card)를 이용한 로봇의 속도 및 위치 제어는 모션 보드내의 주연산장치인TMS320C31을 이용하여 행해진다. 상위 제어기(OS)는 일정 주기(16ms)마다의 명령을 모션보드내의 DP-Ram 영역에 저장을 하고 모션 보드는 그것을 1ms마다 발생되는 인터럽트 과정에서 명령을 읽어오도록 제작하여 상위제어기에서 명령을 입력하는 즉시 수행될 수 있도록 하였다. 이렇게 입력 받은 명령을 모션 보드는 내부에서 처리하고 속도제어 모드로 지정된 디지털 서보에 모션보드 내부의 DAC를 통해 DC값을 출력한다. 서보 드라이브는 인가된 전압에 비례하여 서보 모터의 속도를 제어하여 상위 제어기에서 입력된 명령이 처리되도록 설계하였다. 아래 그림 4는 모션보드의 외관 모습이다. 추가가적으로 이 모션보드에는 32개의 I/O Port가 내장 되어 있어 모션제어이외에 추가적으로 I/O제어가 가능하다.

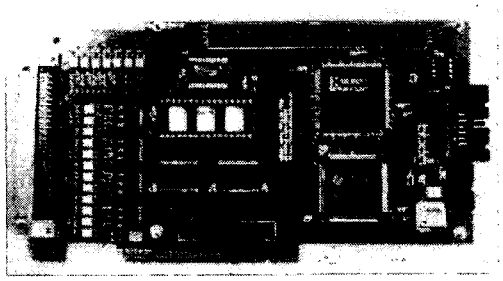

그림 4 모션보드의 모습

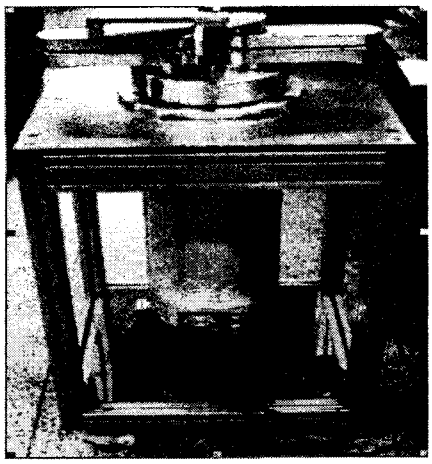

그림 5 반도체 이송장비용 로봇

5.3 로봇

로봇은 아이램테크의 반도체 이송장비 로봇을 사용하였다. 위의 그림 5는 사용된 로봇의 외관이다.

6. GUI

본 논문에서의 GUI 프로그래밍은 Trolltech의 QT를 사용하였다. 하지만 상위제어기의 로봇코어는 그것의 인터페이스 역할을 하는 공유메모리상의 프로토콜을 만족한다면 어떠한 Graphic Library를 사용해도 무관하다. 아래 그림 6은 QT를 이용해서 제작한 로봇제어 GUI의 외관 모습이다.

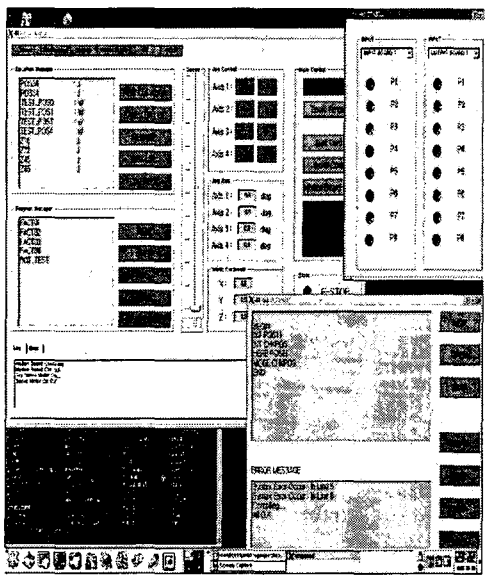

그림 6 로봇제어용 GUI

7.결론 및 향후 연구과제

Linux는 현재 연구용뿐만 아니라 상업용에서 많은 적용이 이루어지고 있다. Linux의 가치는 비용이 적게 들어가는 장점이외에 연구적이 가치가 많은 OS이다. 실제로 Real-Time Linux로 개발을 해본 결과 아직은 상용 Real-Time OS에 비해 개발환경 및 성능등이 뛰어나지는 않았다. 하지만 개발 비용이 비싸다는 상용 RTOS의 특성을 염두 한다면 Linux자체는 그만큼 가치를 가지게 된다. 아직까지 제어분야에 많은 적용이 이루어지지 않은 현 상황에서 그 적용 가능성을 로봇제어라는 주제를 통해 제시해 보았다. 앞으로의 향후과제로 현재 4축의 로봇만을 컨트롤 할수 있는 것을 6축까지 확장하고 여러 종류의 로봇을 컨트롤 할 수 있는 것을 목표로 한다. 따라서 소스를 오픈 함으로서 로봇에 관한 개발을 하고있는 개발자들을 유도해 여러 형태의 로봇의 데이터를 라이브러리화 시켜 사용할수 있게 하는 것이 최종적인 목표가 될것이다.

참고 문헌

[1] E.Bianchi, L. Dozio, G.L. Ghiringhelli, P. Mantegazza, Complex Control Systems, Applications of DIAPM-RTAI at DIAPM, Realtime linux Workshop Vienna 1999

[2] E.Bianchi, L. Dozio, Some Experiences in fast hard realtime control in user space with RTAI-LXRT, Realtime Linux Workshop Orlando 2000

[3] P.Cloutier, P.Mantegazza, S. Papacharalambous, I. Soanes, S. Hughes, Karim Yaghmour. DIAPM-RTAI POSITION PAPER, NOV 2000

[4] G.Quaranta, P. Mantegazza, Using MATLAB-Simulink RTW to Build Real Time Control Applications in User Space with RTAI LXRT, Realtime Linux Workshop Milano 2001

[5] Matthias Kalle Dalheimer, Programming with Qt, 2002

[6] John R. Levine, Tony Mason, Doug Brown, lex & yacc, 1995