2003년도 대한전자공학회 하계종합학술대회 제26권 제1호

Embedded linux kernel 을 이용한 효율적인 모바일 단말 구현에 관한 연구

이용훈, 윤원동, 김영근
삼성전자 디지털 미디어 연구소
Platform Goup,Mobile platform lab

The research on embody in mobile system efficiently using Embedded linux kernel

Lee Yong Hoon, Yun Won Dong, Kim Young Kuen

Mobile Platform Lab, Platform Group
Samsung Electronics Co., Ltd
E-mail : sanghanp.lee@samsung.com

Abstract

본 논문에서는 PDA, Hand PC(HPC)등과 같은 모바일 단말에 운영체제로써 Embedded Linux 를 채택하였을 경우 고려해야 할 Hardware 사양, I/O interrupt latency 에 따른 성능, 스케쥴링 정책에 따른 성능에 대하여 논한다. 대상 타겟으로 사용한 HPC 의 하드웨어 사양에 설명하고, Embedded Linux 와의 연동에 있어서 문제점을 살펴 본다. 또한 각종 I/O device 들의 Interrupt latency 에 따른 성능저하와 스케쥴링 정책에 의한 성능저하에 대하여 분석하고, 해결 방안에 대하여 논한다. 마지막으로 실제 예로서 Mobile IPv6 S/W Stack 을 이용한 실제 검증을 수행하고 성능 향상 방안을 제시한다.

I. 서론

PDA, HPC 와 같은 모바일 단말의 운영체제로 Embedded Linux 를 채택할 경우 우선적으로 하드웨어의 사양을 고려하여야 한다. 일반 Embedded 시스템은 제한된 자원(CPU, 메모리, I/O device)을 제공하는데, Embedded Linux 는 다른 상용 RTOS(Real-Time Operating System)보다 높은 하드웨어 사양을 요구한다. 우선 Embedded Linux 를 동작 시킬 수 있는 최소한의 하드웨어 사양, 특히 CPU,휘발성 메모리, 비휘발성 메모리, I/O device, Power Supplier 를 제시하고 성능을 분석 할 것이다.

Interrupt-Driven I/O 에 있어서 Embedded linux 의 ISR 의 코딩 형태로 인하여 오버헤드가 발생, I/O device 에서 보낸 Interrupt signal 을 CPU 가 놓칠 확률이 존재 한다. 따라서, 본 논문에서는 이런 상황이 발생할 확률과 이로 인한 성능 저하의 정도를 알아 보고, 해결 방안을 모색한다.

Linux 시스템은 타임 쉐어링 공정 스케쥴 알고리즘을 사용하고 있다. 엄격한 우선순위 선전(Priority Preemptive)과 실시간에 적합한 스케쥴러를 가지고 있지 않다. 따라서, 본 논문에서는 이러한 스케쥴링 알고리즘이 적용되고 있는 리눅스로 구현한 모바일 단말의 성능과 발생할 수 있는 critical 한 문제들을 알아보고 해결방안을 모색한다.

최종적으로 본 논문에서는 Mobile ipv6 stack 을 단말에 포팅하여 Hand-over 가 발생하는 상황을 재현, 멀티미디어 데이터를 전송받는 실험을 하여 결과를 분석하

여 성능이나 개선 방향을 제시할 것이다.

본 논문의 순서는 다음과 같다. 2 장에서는 Embedded Linux 가 포팅가능한 시스템, 3 장에서는 I/O interrupt latency 에 따른 성능저하 및 개선방안, 4 장에서는 Task scheduling 정책에 따른 성능 및 성능개선방안, 5 장에서는 Mobile IPv6 Stack 을 이용한 시스템 성능 측정, 마지막으로 본 논문의 결론을 낼 것이다.

II. Embedded Linux 포팅 가능한 시스템

일반 PC 는 하드웨어의 영향을 그다지 받지 않지만 Embedded 시스템은 하드웨어 부분에서 고려하여야 할 부분이 많다. 특히나 모바일 단말의 경우는 전력소모 부분까지 고려하여야 한다. 표 1 은 현재 작업중인 HPC 의 하드웨어 사양이다. 표를 보면서 각 컴퍼넌트에 고려 사항을 알아 보자.

항목	내용
CPU 제조사	Intel
CPU 이름	Xscale PXA250 SA1110,400Mhz
Memory	SDRAM64M,삼성 NAND Flash 32M
Serial Port	CPU 내장 FFUART
USB	CPU 내장 Client,Host 동작가능
PCMCIA	PCMCIA 1 port,CF 1 port
PCMCIA to Local bus Interface	SA1111
LCD	Epson
Sound Codec	WM9705

표 1 HPC 사양

우선 시스템을 구성하는데 있어 가장 중요한 요소인 CPU 관련 기본적으로 다음을 고려해 보아야 한다.

1. Bit 수: 16bit,32bit,64bit
2. 속도:얼마나 빠른 클럭 주파수를 제공하는가.
3. 명령어 방식: RISC or CISC
4. 엔디안
5. 캐쉬 메모리
6. 전원 : APM(Advanced Power Management)지원
7. MMU 지원

기본적으로 모바일 단말에 Embedded Linux 를 구동시키기 위해선 32bit 내지 64bit 의 CPU 를 사용하여야 한다. 물론 16bit,8bit 머신에 Embedded Linux 를 포팅하는 프로젝트가 진행되고 있지만 억세스 가능한 메모리 용량과, MMU, 슈퍼바이저 모드, 캐쉬,파이프라인 같은 시스템 성능 향상에 도움을 주는 기능들이 빠져있어 큰 의미가 없는 프로젝트로 판단된다. 특히 다중 사용자와 다중 프로세스를 지원하는 선점형 운영체제인 Embedded Linux 는 프로세스간,커널과 프로세스간 메모리 보호가 필수적 이므로 하나의 프로세스의 메모리 오류가 전체 시스템에 영향을 미치지 않게 하기 위한 MMU 가 필수적으로 필요하다. 커널과 응용 프로그램의 작업 공간인 메모리의 경우 최소 16M 는 지원 되어야 할 것이다. 물론 커널에 포함된 컴퍼넌트와 구동중인 Task 의 양에 따라 16M 이상의 메모리를 지원 하여야 할 것이다.또한 부트로더, 커널이미지와 File system 이 마운트될 비휘발성 메모리는 32M 이상 지원이 되어야 할 것이다. 위에 언급한 사양에 외부와 통신을 위한 I/O device 를 갖추면 Embedded Linux 포팅을 위한 기본적인 사양은 갖추게 된 것이다. 모바일 단말의 경우 I/O device 를 로컬버스에 직접 물리기 보다는 PCMCIA 를 통해 system 과 연결시키는 추세이며 특히 CF(Compact Flash)type 의 Bus 를 많이 사용한다.

III.I/O Interrupt latency 에 따른 성능저하 및 개선방안

Embedded Linux 커널에서 I/O 처리는 Interrupt driven I/O 방식이다. 그러므로 Interrupt 처리시에 발생할 수 있는 Interrupt latency 나 Interrupt handler duration 에 의해 I/O 의 성능이 결정되며, 이는 또한 전체 시스템의 성능과 직결된다. Interrupt latency 란 I/O device 에서 물리적인 Interrupt signal 을 발생한 후부터 ISR 이 시작되기 전까지 걸리는 시간을 말한다. 현재 Embedded Linux 커널의 Interrupt service routine 을 보면 한 Device 에서 Interrupt signal 을 보내면 Interrupt 가 Nested 되는 것을 막기 위해 cli()를 call 함으로 Hardware 로부터의 Interrupt 를 막는다. 그렇게 되면 cli()가 call 된 후에 발생한 Interrupt 는 Interrupt 를 허용하는 sti()가 call 될 때까지 기다려야 하며,이는 Interrupt 반응속도 즉 Interrupt latency 를 길게 하는 요인이 된다. Interrupt handler duration 은 Hardware 에서 CPU 로 Interrupt signal 을 보낸 후 CPU 의 제어 권한이 Interrupt service routine 으로 넘어간 후 Interrupt service routine 이 끝날 때까지의 걸리는 시간을 의미한다.물론 이 duration 시간이 길수록 Interrupt latency 는 길어 질 것

이고,cli() call 로 인하여 I/O device 에서 보낸 Interrupt signal 을 놓칠 확률이 발생하여 전체적인 시스템의 성능저하를 초래한다.다음은 Embedded Linux 의 Ethernet device 의 driver 의 ISR 의 예이다.

```
Ether_ISR()
{
    ...
    cli();                          /* disable interrupt */
    get_data_from_eth_chip();       /* data processing */
    pass_data_to_stack();
    sti();                          /* enable interrupt */
}
```

위의 코드를 보면 ISR 에서 cli()로 모든 인터럽트를 Disable 시킨 후 Chip 에서 Data 를 가져오는 루틴과 가져온 Data 를 상위 Layer 의 Stack 에 올려주는 루틴을 수행한다. 이런 코딩 형태에서 문제점은 cli()와 sti()사이에 여러 번의 다른 Interrupt 가 발생기에 충분히 길다는 것이다.

모바일 단말인 HPC 에서 이런 형태의 ISR 은 현재까진 큰 문제를 발생시키진 않고 있다. 이는 Processor 의 클럭이 200Mhz 나 되기 때문에 I/O device 의 Data rate 에 비해 빠른 속도로 명령어를 처리할 수 있기 때문이다. 즉,하드웨어의 고급사양으로 문제를 해결 할 수 있었던 것이다. 만일 낮은 클럭의 CPU 를 사용한다면 RTLinux 를 사용하여 문제의 해결 방안을 모색하여야 할 것이다.

IV.Task scheduling 정책에 따른 성능 및 성능개선 방안

Embedded Linux 는 Time-sharing scheduling 을 한다. 즉, 하나의 프로세스를 실행하고 있는데,이 프로세스가 정해진 Time-slice 를 모두 채우거나, Blocking 되어 중단되었을 경우 등을 제외하고는 이 프로세스는 중단되지 않는다. 따라서 지금 수행하고 있는 프로세스보다 높은 Priority 를 가지는 프로세스가 있어도 스스로 CPU 를 놓아주거나 Time-slice 가 끝날 때까지 기다려야 한다. 즉 지금 수행하고 있는 프로세스 중 가장 중요한 프로세스를 수행하고 있다고 보장할 수 없다. 또한 Demand paging 기법을 사용하고 Swapping 을 하기 때문에,프로세스 수행도중 페이지를 디스크에서 메모리로 읽어 들이는 일이 발생하여 프로세스의 수행시간을 예상할 수 없게 할 수 있다. 가상 메모리는 프로세스가 관리하여야 하는 자료구조가 많아지게 하기 때문에 Task switching 이 발생하는 경우 처리 해야하는 일이 많아지기 때문에 필연적으로 성능 저하를 가져온다. Embedded Linux 에서는 또한 동기화가 필요한 부분에서 성능을 위하여 자원을 오랫동안 점유한다. 실제 자원을 필요한 순간에 쓰고 반납하게 되면 자원을 얻고 반납하는 횟수가 늘어나기 때문에 성능이 떨어지게 된다. 그래서 이 횟수를 줄이고 자원을 오랫동안 독점하게 되는데 이를 Coarse grained synchorization 이라고 한다. 또한 Priority 가 낮은 프로세스가 점유하는 자원을 Priority 가 높은 프로세스가 기다려야 하는 Priority inversion 문제도 발생한다. 현재 Preemption patch 와 Real-time patch 가 위의 문제 해결 방안으로 제시되고 있다. Preemption patch 의 기본 Idea 는 커널 task 인 schedule()를 더 자주 동작하게 하는 것과 발생 특정 Event 가 발생하였을 때 schedule() task 가 동작 할때까지의 시간을 최소한으로 줄인 것이다. Real-time patch 는 기존 Embedded Linux 아래에 RTlinux 커널을 올리고 기존 Embedded Linux 커널을 RTlinux kernel 에서 돌아가는 하나의 프로세스로 만들어서 Real-Time Task 와 기존의 Embedded Linux 커널이 공존하는 형태이다. 현재 우리 HPC 에는 Task scheduling 에 따른 Critical 한 문제가 발생하지 않아 위의 2 개의 해결 방안에 대한 patch 는 하지 않은 상태이다. 문제미 발생 이유는 빠른 CPU 클럭이다.

V.Mobile IPv6 Stack 을 이용한 시스템 성능 측정

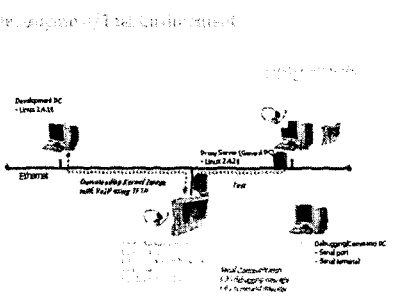

그림 1 Mobile IPv6 Test Lab

그림 1 은 Embedded Linux 가 porting 된 HPC 의

성능 측정을 위한 Test lab 이다. 실험의 내용은 서버(일반 PC)에 있는 음악 MP3 file 을 클라이언트(HPC)에 전송하면서 실시간으로 음악을 Play 해주는 상황에서 Mobile IPv6 hand-over 를 발생시켜 음악의 끊김이나 음질의 정도로 성능을 측정하는 내용이다. Test 를 위해 Embedded Linux 에서 제공하는 Mobile IPv6 Stack 을 HPC 에 포팅 하였고,CF(Compact Flash) Wireless Lan card 를 I/O device 로 채택하였다. Test lab 의 환경은 AP(Access Point)가 내부 Lan 에 연결되어 서버와 클라이언트를 연결 시켜주는 역할을 하게 설정 하였다. 결과적으로 고품질의 음악을 들을 수 있었다. 이는 RTlinux 의 patch 가 없는 Embedded Linux 커널이라도 CPU 의 클럭이 200Mhz 이상 된다면 일반 모바일 단말로서의 기능은 충분히 할 수 있다는 결론을 얻을 수 있다.

VI. 결론

본 논문에서는 모바일 단말의 운영체제로 Embedded Linux 를 채택했을 고려 해야 할 hardware 사양,I/O interrupt latency & interrupt handler duration, Task scheduling 정책, 마지막으로 Mobile IPv6 stack 을 통한 실증을 해 보았다. 향후 과제로는 일반 Embedded Linux 커널에 Real-time 기능을 추가한 RTLinux 에 대하여 분석하여 보고, 모바일 단말에 채택 하였을 경우 유용성 여부에 대한 판단을 할 예정이다.

참고문헌

[1] Measuring Interrupt Latency within the FreeBSD kernel
Sansonetti Laurent 3rd December 2001

[2] Performance Advantages for Embedded Systems
TimeSys Linux GPL

[3] 임베디드 리눅스
박재호 한빛미디어

[4] Embedded Linux – Ready for Real-time
Bill Weinberg, Montavista Software Inc.

[5]Linux Lab Notes
권수호 삼성전자 CTO