휴대폰용 임베디드 리눅스 부팅시간 단축기법*

이종일° 남영진† 김성률‡ 서대화‡
경북대학교 정보통신공학과, †대구대학교 컴퓨터·IT공학부
‡임베디드소프트웨어 협동연구센터
sysnet79@gmail.com°, yjnam@daegu.ac.kr, srkim@escrc.re.kr, dwseo@ee.knu.ac.kr

Techniques for Shortening the Boot-time of Embedded Linux for Mobile Phones

Jongil Lee°, Young Jin Nam†, Sung-Ryul Kim‡ Dae-Wha Seo‡
Dept. of Information & Communication Engineering, Kyungpook National University
†School of Computer & Information Technology, Daegu University
‡Embedded Software Cooperative Research Center

요 약

휴대폰에 임베디드 리눅스를 탑재하고자 할 경우 전력소모, 경량화, 빠른 부팅, 실시간, 보안 등과 관련한 추가고려사항이 발생한다. 본 연구에서는 휴대폰용 임베디드 리눅스의 커널 부팅시간을 단축시키기 위한 구체적인 방법을 제시하고 실제적인 적용을 통하여 그 성능을 평가한다. 측정된 데이터를 바탕으로 커널 내부에서 가장 많은 시간을 소요하는 함수들을 찾아내고 확인된 함수들에 대해서 초기화 지연회피, 불필요한 장치 드라이버 제거, 불필요한 커널 메시지 미출력 등의 기법들을 각각 적용한다. 제안된 기법을 모두 적용할 경우에 기존 커널 부팅시간을 50%정도 단축시킬 수 있었다.

1. 서 론

휴대폰의 하드웨어 성능이 증가함에 따라서 DMB, 동영상 재생 등과 같은 다양한 멀티미디어 기능을 가지는 스마트 폰의 형태로 발전해 나가고 있다. 또한, 모바일 폰에 다양한 기능들을 신속히 제공하기 위해서는 휴대폰에 탑재되는 운영체제가 오픈 아키텍처를 가져야 한다는 결론에 도달하게 된다. 휴대폰용 임베디드 운영체제 시장은 현재 심비안이 장악하고 있고 그 뒤를 마이크로소프트가 추격하고 있는 양상이다. 리눅스 휴대폰은 (임베디드) 리눅스를 휴대폰 환경에 맞게 재조정하는 작업에 상당 기간이 소요되어 보급 속도가 기대보다는 느리지만 지속적인 성장추세를 보여주고 있다. IDC 조사에 따르면 2006년까지는 리눅스가 고성능 스마트 폰용 소프트웨어 시장의 약 4.2%를 차지할 것으로 추정되고 있다.

휴대폰 운영체제에 공개 플랫폼인 리눅스를 사용함으로써 이동 통신 사업자와 휴대폰 단말기 제조업체, 그리고 응용 소프트웨어 개발업체와 관련 연구자들은 다양한 장점을 얻을 수 있다. 이는 휴대폰 운영체제 시장에서 임베디드 리눅스의 보급을 지속적으로 증가시키는 원동력이 되고 있다. 리눅스는 오픈소스 기반으로 개발되어 휴대폰 단말기 제조업체들이 자유롭게 사용할 수 있고 필요에 따른 수정이 가능하며, 심비안이나 마이크로소프트 운영체제보다 대당 5~7달러 정도 소프트웨어 가격이 저렴하기 때문에 비용을 절감할 수가 있다. 또한, 리눅스는 시계와 같은 간단한 제품에서 컴퓨터 같이 복잡한 제품에 이르기까지 광범위한 활용이 가능하기 때문에 다양한 디자인을 창출할 수 있으며, 여러 가지 다양한 조합이 가능하다. 따라서, 휴대폰 단말기 제조업체들은 부가적인 기능을 추가하여 차별화된 단말기를 다양하게 생산할 수 있는 장점을 가진다. 응용 소프트웨어 개발업체들은 기존의 폭넓은 리눅스용 소프트웨어 인프라를 활용할 수 있다. 리눅스는 공개 플랫폼으로서 로열티가 없고 소스코드가 공개되어 있어 프로그래머가 원하는 대로 특정 기능을 추가하거나 삭제할 수 있으며 다양한 플랫폼에 포팅이 가능하다. 리눅스는 기존의 임베디드 운영체제 시장에서 우위를 점하고 있던 RTOS의 지원을 받은 GNU 툴에 기반하고 있어 수십만에서 많게는 수백만에 이르는 GNU 툴에 익숙한 개발자를 확보할 수가 있다.

휴대폰 운영체제로 리눅스가 사용되기 시작하면서 곳곳에서 휴대폰을 위한 임베디드 리눅스를 연구하는 공개된 워킹그룹이 활동을 시작하여 체계를 잡아가고 있다. OSDL(Open Source Development Labs)은 오픈 소스 운영체제인 리눅스를 휴대폰에서 이용할 수 있도록 하는 Mobile Linux Initiative(MLI) 워킹그룹을 개설하였다. MLI 워킹그룹에서는 어플리케이션 개발을 추진하고 서로 다른 휴대폰에서 필요한 사항에 대해 표준화를 진행하며 관련 오픈 소스 개발 프로젝트를 운영할 계획이다. 가전기기를 위한 리눅스 포럼인 Consumer Electronics Linux Forum(CELF)은 휴대폰 기능을 위한 API를 개발하는 워킹그룹인 Mobile Phone Profile Working Group을 개설하여 Mobile Phone API 초안을 지속적으로 발표하고 있다. 또한, CELF는 포럼 내에 부팅시간, 전력관리, 실시간, 보안, 시스템 크기와 같은 휴대폰에 적용 가능한 리소스와 프로젝트를 다루고 있다. 2005년 11월에는 ARM, 프랑스텔레콤/오렌지, 미지리서치 등 세계적인 주요 통신 관련 회사들이 참여해 무선 단말기용 리눅스 표준화를 추진하는 LiPS(Linux Phone Standard) 리눅스 폰 표준화 포럼이 출범하여 활동을 시작하였다. LiPS는 리눅스 폰의 개발 및 호환성에 영향을 미치는 서비스와 API를 표준화해 이를 유/무선 및 복합 단말기에 적용하는데 초점을 맞추고 있다.

휴대폰용 임베디드 리눅스는 데스크 탑용 리눅스와 달리 몇가지 부가적인 기능을 필요로 한다[1]. 첫째, 배터리 재충전 없이 장시간 동안 휴대폰을 사용할 수 있도록 하는 저전력(low power) 기능이 제공되어야 한다. 둘째, 휴대폰 내에는 일반적으로 16~32MB 크기의 메인 메모리가 존재한다. 이는 데스크 탑에 존재하는 메인 메모리의 크기가 256MB~1GB 정도임을 고려할 때에 상대적으로 매우 적다는 것을 알 수 있다. 휴대폰에서 메인 메모리 및 플래시 메모리 크기는 장치 가격에 큰 영향을 미치는 요소이기 때문에 신중하게 그 크기가 결정되어져야 한다. 따라서, 휴대폰에 저장/실행되는 커널 및 응용프로그램의 크기가 되도록 작은 공간을 차지하도록 설

* 본 연구는 경북대 임베디드S/W협동연구센터 협동연구개발과제와 정보통신부 및 정보통신연구진흥원의 대학IT연구센터지원사업의 연구결과로 수행되었음 (IITA-2005-C1090-0501-0018)

계되어야 한다. 셋째, 휴대폰에 전원버튼을 누른 후에 통화할 수 있는 상태까지 되는데 되도록 짧은 시간이 소요 되도록 커널 및 응용 프로그램에 튜닝되어야 한다. 일반적으로 KDE를 사용하는 리눅스 노트북이 부팅되는 시간이 63초 정도인 것에 반해서 리눅스가 탑재된 가전제품의 경우에 부팅시간을 최대 10초 정도도 잡고 있다[1]. 휴대폰도 가전제품과 마찬가지로 최대 10초 정도의 부팅시간을 적당한 수준으로 보고 있다. 부가적으로, 휴대폰용 리눅스에서는 다양한 응용들이 동시에 실행되고 네트워크 형태로 접속됨에 따라서 주어진 시간 내에 일을 처리할 수 있는 실시간성(real-time)과 외부의 다양한 악의적인 공격에 대처할 수 있는 보안(security)기능의 중요성이 대두되고 있다.

본 논문에서는 휴대폰용 임베디드 리눅스에서 필요한 여러 가지 기능들 중에서 부팅시간을 최소화 하는 기법에 대한 연구를 목적으로 한다. 특히 커널내부에서 소모되는 시간을 단축할 수 있는 방안에 대해 초점을 맞추었다. CELF의 BTWG(Boot Up Time Working Group)에서 빠른 부팅을 위하여 프로세서의 주파수 계산을 줄이는 초기화 지연회피 방법, 사용되지 않는 장치들에 대한 불필요한 장치구동기 제거, 불필요한 커널 메시지 미출력 등의 여러 가지 방법들을 제시하고 있다. 이밖에도 경량화를 위한 방법 중의 하나인 XIP(execute-in-place)기법을 적용할 수 있으며, 장치 구동기 초기화를 지연시키는 방법 등의 다양한 방법들이 존재한다. 본 논문은 다음과 같이 구성되어 있다. 2장에서는 배경지식으로 모바일 폰용 임베디드 리눅스와 부팅순서, 빠른 부팅을 위한 CELF 명세를 기술한다. 3장에서는 커널 부팅시간을 측정하기 위한 기법과 본 연구에서 제안하는 커널 부팅시간 단축 기법과 실험결과를 기술한다. 끝으로 4장에서는 본 연구에 대한 요약 및 향후 연구과제에 대해 기술한다.

2. 배경지식

2.1 임베디드 리눅스 부팅과정

리눅스가 부팅시 소요되는 시간을 단축시키기 위해 먼저 리눅스의 부팅과정을 분석하였다. 리눅스의 부팅과정은 3가지의 주요 단계로 분류할 수 있다. 펌웨어(firmware)단계, 커널 단계, 사용자 스페이스(space) 단계이다[2]. 전원이 인가되면 플래시 메모리 내에 압축되지 않는 상태로 저장되어 있는 부트로더가 실행되면서 플래시 메모리 내에 압축된 커널 이미지를 램 영역에 복사한다. 복사된 압축된 커널 이미지는 램에서 압축을 해제하는데 이를 펌웨어단계 혹은 부트로더 단계라고 한다. 커널이 시작되어 사용자 스페이스가 시작되기 전까지의 과정을 커널 단계라고 한다. 사용자 스페이스가 시작된 후 RC 스크립(script)이 시작되어 시스템에 필요한 서비스들을 구동시킨 후 어플리케이션을 시작함으로써 사용자가 시스템을 사용할 수 있도록 하는 단계를 사용자 스페이스 단계라고 한다. 이러한 단계별 정의는 전원인가, 부트로더, 커널로딩, 커널초기화, 사용자 스페이스 초기화의 순서로 정의하기도 한다[2]. 특히, 본 연구에서는 사용자 스페이스 단계가 시작되기 전까지의 소요되는 시간을 줄이는데 초점을 맞춘다.

2.2 CELF(Consumer Electronics Linux Forum) 명세

CELF은 2003년 6월에 가전분야에서 활약 중인 몇몇 전자업체가 연합해서 임베디드 리눅스 플랫폼표준을 구축하려는 목적으로 소니와 마쓰시다가 연합해서 설립한 단체이다. 현재 삼성, LG, ETRI를 비롯한 국내업체와 ARM, HP, IBM, 몬타비스타를 비롯한 범세계적인 회사와 기관이 CELF에 참여하고 있다. CELF 명세[3]는 가전용 전자제품을 개발하는 과정에서 사용할 리눅스기반 구조를 더 좋게 만들기 위해 필요한 기능과 기술을 정의하고 지정하려는 목적으로 만든 문서이다. 부트업 시간명세는 BTWG(Boot Up Time Working Group)에서 진행하고 있으며 현재 버전 1.0R2까지의 주요 내용은 초기화 지연회피, IDE 장치 검색회피, 커널 XIP(execute-in-place)등의 세가지 이슈를 다루고 있다. 본 논문에서는 CELF 명세를 기본적으로 직접적으로 적용한다.

2.3 커널내부 함수 소모시간 측정방안

CELF는 커널 내부의 성능평가와 문제해결을 위해 커널내부 함수들이 소모하는 시간을 측정하는 도구인 KFI(Kernel Function Instrumentation)[4]를 제공한다. CELF에서 제공하는 KFI는 x86기반에서만 동작하도록 현재 개발되어 있다. 본 논문에서는 기존 KFI를 소스를 부분적으로 수정하여 ARM(PXA-255) 환경에서 동작할 수 있도록 하였다. KFI가 x86환경에서는 `rdtsc`를 이용하여 내부 오실레이터가 제공하는 매우 높은 시간 정밀도를 제공하고 있는 반면에, 현재 ARM에서 동작할 때는 내부 스케줄러가 사용하고 있는 클록 해상도인 10msec를 이용하고 있다.

3. 제안된 커널부팅시간 단축기법

본 절에서는 ARM기반의 플랫폼에서 부팅시 소요되는 시간에 대한 분석결과와 이러한 결과를 바탕으로 시간 소요가 많은 루틴들로 인한 커널의 부팅시간을 단축할 수 있는 다양한 기법에 대해 기술한다.

3.1 휴대폰용 임베디드 리눅스 환경

본 논문에서 제안한 기법들을 시험하기 위한 휴대폰용 임베디드 리눅스 환경은 다음과 같다. 휴대폰 하드웨어 환경으로 CDMA 무선 통신 모듈이 부착된 T-Box Xt 임베디드 보드[5]를 이용하였다. Intel PXA255 Xscale 프로세서, Intel StrataFlash E28F128 NOR 플래시 메모리 32M(16MB×2), 삼성 K4S561632 SDRAM 64M(32MB×2), 3.5인치 해상도 240×320 TFT LCD가 장착되어 있다. 사용된 임베디드 리눅스는 커널 버전 2.6.8.1에 PXA-255 Xscale 프로세서와 T-Box용 보드용 소스코드가 패치 되었다. 또한, 커널 내에 존재하는 각 함수에서 소요되는 시간을 측정하기 위해서 KFI 코드를 추가적으로 패치하였다.

3.2 부팅시 소요되는 시간분석

부팅에 소요된 총 커널시간은 12.45초로 측정되었다. KFI를 이용하여 커널내부에서 가장 많은 시간을 소모하는 상위 150여개의 함수목록을 정리하였으며, 함수 호출횟수와 소요시간을 분석한 결과 두 가지의 유형으로 분류할 수 있었다. 첫째, 표1 에서와 같이 프로세스 전환에 사용되는 `schedule`함수와 IRQ 처리를 위한 `pxa_timer_interrupt`함수 등과 같이 함수 내부의 소요시간은 짧은 반면에 호출횟수가 비교적 많아 전체적인 소요시간이 큰 함수이다. 하지만, 이 부류의 함수들에 의해서 소요되는 시간을 줄이기는 거의 불가능하다는 것을 쉽게 유추할 수 있을 것이다.

표 1. 높은 호출 횟수로 소요시간이 긴 함수 (단위:us)

함수명	호출횟수	소모시간	평균소모시간
schedule	277	4,1730,000	150.649
do_level_IRQ	1,246	1,2460,000	10.000
__do_irq	1,246	1,2460,000	10.000
asm_do_IRQ	1,246	1,2460,000	10.000
pxa_timer_interrupt	1,245	1,2450,000	10.000
timer_tick	1,245	1,2450,000	10.000
do_timer	1,245	1,2450,000	10.000

둘째, 표 2 에서와 같이 NAND타입의 플래시 메모리를

제어하는 MTD(Memory Technology Device)[6] API함수와 IDE장치들과 관련한 함수등과 같이 호출횟수는 적은 반면에 함수 자체의 소모시간이 긴 함수이다. 특히, NAND 플래시 메모리 관련 함수에서 소요되는 시간이 약 4초정도로 매우 긴 것을 볼 수 있다.

표 2. 단일 호출 소요시간이 긴 함수 (단위:us)

함수명	호출횟수	소모시간	평균소모시간
nand_scan	1	4,810,000	4,810,000
nand_memory_bbt	1	4,790,000	4,790,000
nand_default_bbt	1	4,790,000	4,790,000
nand_scan_bbt	1	4,790,000	4,790,000
create_bbt	1	4,790,000	4,790,000
nand_read_raw	477	4,770,000	10,000
ide_init	1	870,000	870,000
ide_register_hw	1	860,000	860,000
init_ide_data	1	860,000	860,000
ide_arm_init	1	860,000	860,000

이외에도 표 3 에서와 같이 CELF BTWG명세에서 언급하였던 calibrate_delay함수와 초기부팅정보를 디스플레이하는 콘솔 관련 함수인 printk함수도 확인할 수 있었다.

표 3. 함수들의 호출 횟수와 소모시간 (단위:us)

함수명	호출횟수	소모시간	평균소모시간
printk	42	420,000	10,000
calibrate_delay	1	200,000	200,000

3.3 제안된 기법 및 성능평가 결과

● 초기화 지연회피 (preset-LPJ)

리눅스는 시스템을 초기화할 때 마다 클록신호의 주파수를 알아야한다. 커널을 컴파일 할 때 이 주파수를 선언하지 않기 때문에 똑같은 커널 이미지를 클록이 다른 프로세서에서 실행할 수 있다. 대기를 위해 필요한 지연시간인 loop_per_jiffy를 구하기 위해 calibrate_delay함수가 사용된다[7]. 부팅과정에서 이 함수가 차지하는 시간이 부팅 지연에 상당한 영향을 미친다. 이런 문제점을 해소하기 위해 부팅과정에서 매번 loop_per_jiffy를 계산하는 대신에 미리 계산한 고정 값을 사용하여 함수로 인한 부팅시 소요되는 시간을 단축할 수 있다. 본 논문에서는 lpj값을 392.39 BogoMips로 설정하여 부팅시간을 26msec 정도 단축하였다.

● 불필요한 장치 드라이버 제거 (no-ide, no-nand)

부팅 시 특정장치를 탐색하지 않게 할 경우 부팅에 소요되는 시간을 개선할 수 있다. 임베디드 장치의 특성상 특정 장치만을 사용하며 추가되는 장치가 없기 때문에 특정 장치가 사용되지 않는다면 커널에서 해당 드라이버를 제거하거나 추후 모듈로 적재하여 부팅시 소요되는 시간을 단축시킬 수 있다. 모바일 폰에 사용되지 않는 대표적인 장치 드라이버인 IDE관련 장치를 들 수 있다. 또한, NAND 플래시의 경우에도 현재까지 휴대폰에서 선택적으로 이용되고 있다. 본 논문에서 사용한 T-Box xt 임베디드 보드에서는 NAND 플래시 메모리가 장착은 되어 있지만, 커널/응용에서 이용하지 않고 있다. 즉, IDE 관련 장치 드라이버를 제거하여 약 1.31초정도의 시간을 단축시킬 수 있었으며, NAND관련 장치 드라이버를 제거하여 4.25초 정도의 부팅시간을 단축시킬 수 있었다.

● 불필요한 커널 메시지 미출력 (quiet)

커널에서 사용하는 표준출력 함수인 printk함수는 부팅 시 커널 내부에서 일어나는 로그들을 콘솔로 내보내는 역할을 수행한다. 그러나 printk함수는 많은 지연을 가지고 있기 때문에 사용자가 반드시 시스템적인 해당 로그를 확인할 필요가 없으므로 이를 제거하여 63msec 정도 부팅시간을 단축할 수 있었다.

● NAND 플래시 메모리 초기화 지연

커널과 응용에서 데이터 저장을 위한 공간으로 NAND 플래시를 이용할 경우에는 초기화 작업을 진행해야 한다. 초기화시에 소요되는 대부분의 시간은 NAND 플래시 메모리 내에 존재할 수 있는 불량 블록을 검출하여 테이블 형태로 만드는 작업을 수행하기 위해서 소요된다. NAND 플래시 메모리의 크기가 커질수록 이 초기화에 소요되는 시간은 선형적으로 증가하게 되며, 커널 부팅시간에 직접적인 영향을 준다. 이 문제를 해결하기 위해서는 NAND 플래시 메모리 초기화 시점을 커널 부팅이후로 지연하는 방법을 이용할 수 있다[8].

끝으로, 표 4 는 각 제안기법을 적용했을 때, 개선되는 부팅시간과 기존 커널부팅시간에 대한 개선율을 요약하여 보여주고 있다. 제안된 기법들을 모두 적용할 경우, 기존 부팅시간에 비해서 약 45%이상의 단축효과가 있었다.

표 4. 제안된 기법별 단축시간 (단위:us) : 기존 커널부팅시간 12.45초

제안된기법	quiet	preset-LPJ	no-ide	no-nand	합계
단축시간	63,000	26,000	1,310,000	4,250,000	5,649,000
개선율(%)	5.07%	2.09%	10.53%	34.14%	45.31%

4. 결론 및 향후 과제

휴대폰에서 임베디드 리눅스를 사용하고자 할 때는 기존 데스크 탑에서와 달리 전력소모, 경량화, 빠른 부팅, 실시간, 보안 특성 등에 대해서 추가적으로 고려해야 한다. 본 연구에서는 특히 앞서 살펴본 임베디드 리눅스의 주요 부팅 단계중 커널 부팅시간을 단축시키기 위한 방안에 대한 기법들을 제시하고 실제적인 적용을 통하여 성능을 평가하였다. KFI를 ARM 환경으로 이식하고, 측정된 데이터를 바탕으로 커널내부에서 가장 많은 시간을 소요하는 함수들을 찾아내었다. 확인된 함수들에 대해서 초기화 지연회피, 불필요한 장치 드라이버 제거, 불필요한 커널 메시지 미출력 등의 기법들을 적용함으로써 커널 부팅시간을 45%이상 단축시킬 수 있었다. 본 연구에 뒤이어 사용자 스페이스 단계에서의 부팅시간 단축을 위하여 RC스크립트 최적화와 병렬실행 등에 관한 작업이 현재 진행중이다. 또한, 커널 내부의 함수시간 측정 방안인 KFI가 가지고 있는 30%의 오버헤드를 줄이는 방안과 측정 데이터의 신뢰성을 위하여 현재 ARM 상에서 10ms로 설정된 시간 해상도를 1ms 정도로 개선시키는 작업이 진행 중이다.

참고문헌

[1] Y. Nakamoto, "Technical challenges toward the next generation mobile phone linux," Proc. of Int'l Conf. on Computer & Information Technology, Sep. 2004.

[2] Daniel Plerre Bovet. "Understanding the Linux Kernel" 3rd Ed., O'Reilly, 2005.

[3] CELF Specification V 1.0 R2.

[4] http://tree.celinuxforum.org/CelfPubWiki/KernelFunctionInstrumentation.

[5] TBXt_User_Guide_R1.0k. http://www.palmpalm.com, 2005.

[6] MTD(Memory Technology Device NAND Driver Programming) http://www.linux-mtd.infradead.org/tech/mtdnand/book1.html.

[7] Daniel P. Bovet, "Understanding the Linux Kernel 3rd", O'Reilly 2005.

[8] 남영진, 김성률, 서대화, "지연 NAND 플래시 초기화 기법을 탑재한 휴대용 기기" 임베디드S/W협동연구센터, 내부문서, 2006년 3월.