



(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。 Int. Cl. G06F 9/44 (2006.01) G06F 15/16 (2006.01)		(45) 공고일자 (11) 등록번호 (24) 등록일자	2006년12월15일 10-0658564 2006년12월11일
(21) 출원번호	10-2004-0101946	(65) 공개번호	10-2006-0062939
(22) 출원일자	2004년12월06일	(43) 공개일자	2006년06월12일
심사청구일자	2004년12월06일		

(73) 특허권자	한국전자통신연구원 대전 유성구 가정동 161번지
(72) 발명자	이병복 대전광역시 서구 월평동 백합아파트 104동 703호 노광현 경기도 고양시 일산구 마두동 강촌마을 라이프아파트 522동 1406호 권혜연 대전광역시 유성구 가정동 과기대 교수아파트 13동 304호 박애순 대전광역시 유성구 어은동 한빛아파트 138동 301호
(74) 대리인	유미특허법인
(56) 선행기술조사문헌	JP05224976 A JP10312307 A JP2004013370 A 1020040051276 * * 심사관에 의하여 인용된 문헌
	JP08286948 A JP2000010872 A US20030225567 A1 16013370 *

심사관 : 윤혜숙

전체 청구항 수 : 총 9 항

(54) 임베디드 시스템의 소프트웨어 개발 및 실행 방법, 그리고 이를 이용한 이동 단말 장치

(57) 요약

본 발명은 임베디드 시스템의 소프트웨어 개발 및 실행 방법, 그리고 이를 이용한 이동 단말 장치에 관한 것이다.

본 발명에서는 소프트웨어 개발 단계에서 수정된 프로그램 바이너리 코드를 임베디드 시스템의 플래시 메모리에 기록하지 않고 분산 네트워크 파일 시스템을 이용하여 임베디드 시스템에서 실행시킨다. 즉, 수정된 프로그램 바이너리 코드를 호스

트 컴퓨터의 분산 네트워크 파일 시스템에 직접 반영하고, 이후 로컬 네트워크를 통하여 호스트 컴퓨터에 접속되는 임베디드 시스템의 루트 파일 시스템의 서브 디렉토리에 마운트된 분산 네트워크 파일 시스템이 상기 프로그램 바이너리 코드를 읽어와서 실행시킨다.

이러한 본 발명에 따르면 임베디드 시스템의 소프트웨어 개발 완료 단계에서 최적화된 프로그램 바이너리 코드만을 임베디드 시스템의 플래시 메모리에 기록할 수 있으므로, 불필요한 플래시 메모리로의 코드 기록 횟수를 감소시킬 수 있다.

대표도

도 1

특허청구의 범위

청구항 1.

로컬 네트워크를 통하여 임베디드 시스템, 호스트 컴퓨터, 콘솔 장치, 그리고 개발자 장치가 연결되어 있고, 상기 임베디드 시스템의 소프트웨어를 개발하고 실행시키는 방법에서,

- a) 상기 호스트 컴퓨터로부터 부트로더 파일, 커널 파일, 루트 파일 시스템에 대한 이미지 파일들을 제공받아, 상기 임베디드 시스템의 플래시 메모리에 저장하는 단계;
- b) 상기 임베디드 시스템의 플래시 메모리에 저장된 부트로더 파일 및 커널 파일의 실행에 따라 커널 자동 부팅이 수행되는 단계;
- c) 상기 커널 자동 부팅 수행에 따라 상기 플래시 메모리에 저장된 루트 파일 시스템이 임베디드 시스템의 루트에 마운트되는 단계;
- d) 상기 임베디드 시스템의 요청에 따라 상기 호스트 컴퓨터의 분산 네트워크 파일 시스템(이하, 제1 분산 네트워크 파일 시스템이라고 명명함)이 상기 루트 파일 시스템의 서브 디렉토리에 마운트되는 단계;
- e) 상기 임베디드 시스템에 마운트된 분산 네트워크 파일 시스템(이하, 제2 분산 네트워크 파일 시스템이라고 명명함)이 호스트 컴퓨터로부터 전달되는 바이너리 코드 실행 요청에 따라, 상기 호스트 컴퓨터에서 컴파일되어 상기 제1 분산 네트워크 파일 시스템에 저장되어 있는 프로그램의 바이너리 코드를 읽어오는 단계; 및
- f) 상기 임베디드 시스템의 제2 분산 네트워크 파일 시스템이 상기 읽어온 프로그램 바이너리 코드를 실행시키는 단계를 포함하는 임베디드 시스템의 소프트웨어 개발 및 실행 방법.

청구항 2.

제1항에 있어서

상기 a) 단계는

상기 임베디드 시스템이 호스트 컴퓨터로부터 부트로더 파일을 제공받아 플래시 메모리에 이미지 형태로 기록하는 단계;

상기 임베디드 시스템으로의 전원이 차단 및 재공급되어, 상기 플래시 메모리에 기록된 부트로더 이미지가 실행되는 단계;

상기 콘솔 장치로부터 커널 이미지 전송 요청이 입력되면, 상기 임베디드 시스템이 상기 호스트 컴퓨터로 커널 이미지 전송을 요청하는 단계;

상기 요청에 따라 상기 호스트 컴퓨터로부터 제공되는 커널 파일을 상기 플래시 메모리에 이미지 형태로 기록하는 단계;

상기 콘솔 장치로부터 루트 파일 시스템 이미지 전송 요청이 입력되면, 상기 임베디드 시스템이 상기 호스트 컴퓨터로 루트 파일 시스템 이미지 전송을 요청하는 단계; 및

상기 요청에 따라 상기 호스트 컴퓨터로부터 제공되는 루트 파일 시스템 파일을 상기 플래시 메모리에 이미지 형태로 기록하는 단계

를 포함하는 임베디드 시스템의 소프트웨어 개발 및 실행 방법.

청구항 3.

제1항에 있어서

상기 임베디드 시스템의 하드웨어를 메인 메모리, 플래시 메모리, 시리얼 통신 인터페이스, JTAG(joint test action group) 통신 인터페이스, 네트워크 통신 인터페이스, 입출력 장치 통신 인터페이스로 구성하는 단계;

상기 호스트 컴퓨터의 하드웨어를 메인 메모리, 플래시 메모리, 하드디스크, 네트워크 통신 인터페이스, 시리얼 통신 인터페이스, 병렬 통신 인터페이스, 입출력 장치 통신 인터페이스로 구성하는 단계; 및

상기 콘솔 장치의 하드웨어를 메인 메모리, 플래시 메모리, 하드디스크, 네트워크 통신 인터페이스, 시리얼 통신 인터페이스, 입출력 장치 통신 인터페이스로 구성하는 단계

를 더 포함하는 임베디드 소프트웨어 개발 및 실행 방법.

청구항 4.

제1항에 있어서

상기 임베디드 시스템의 제2 분산 네트워크 파일 시스템이 프로그램 바이너리 코드 실행 결과를 상기 콘솔 장치를 통하여 개발자 장치로 전달하는 단계를 더 포함하는 임베디드 소프트웨어 개발 및 실행 방법.

청구항 5.

제4항에 있어서

상기 프로그램 바이너리 코드 실행 결과를 확인한 개발자에 의하여 최종 확정된 프로그램이 상기 콘솔 장치를 통하여 임베디드 시스템의 플래시 메모리에 저장되는 단계를 더 포함하는 임베디드 시스템의 소프트웨어 개발 및 실행 방법.

청구항 6.

제1항 내지 제4항 중 어느 한 항에 있어서

상기 e) 단계는

상기 개발자 장치로부터 소정의 프로그램이 상기 호스트 컴퓨터로 제공되는 단계;

상기 호스트 컴퓨터가 상기 프로그램을 컴파일하고, 컴파일에 따라 생성된 프로그램 바이너리 코드를 제1 분산 네트워크 파일 시스템에 저장하는 단계;

상기 호스트 컴퓨터가 상기 임베디드 시스템으로 바이너리 코드 실행을 명령하는 단계; 및

상기 임베디드 시스템의 제2 분산 네트워크 파일 시스템이 상기 명령에 따라 상기 호스트 컴퓨터의 제1 분산 네트워크 파일 시스템에 저장된 바이너리 코드를 읽어오는 단계

를 포함하는 임베디드 시스템의 소프트웨어 개발 및 실행 방법.

청구항 7.

로컬 네트워크를 통하여 호스트 컴퓨터, 콘솔 장치, 그리고 개발자 장치가 연결되어 있는 이동 단말 장치에서,

상기 호스트 컴퓨터와의 데이터 송수신을 수행하는 제1 인터페이스;

상기 콘솔 장치와의 데이터 송수신을 수행하는 제2 인터페이스;

상기 호스트 컴퓨터로부터 제공받은 부트로더 파일, 커널 파일, 루트 파일 시스템에 대한 이미지 파일이 저장되어 있는 플래시 메모리;

상기 플래시 메모리에 저장된 소정 파일이 로딩되어 구동되는 메인 메모리;

상기 부트로더 파일 및 커널 파일의 구동에 따라 자동 커널 부팅이 이루어지면, 상기 플래시 메모리에 저장된 루트 파일 시스템을 루트에 마운트시키고, 상기 호스트 컴퓨터로부터 분산 네트워크 파일 시스템을 제공받아 상기 루트 파일 시스템의 서브 디렉토리에 마운트시키는 중앙 처리 장치

를 포함하고,

상기 호스트 컴퓨터로부터 전달되는 바이너리 코드 실행 요청에 따라 상기 서브 디렉토리에 마운트된 분산 네트워크 파일 시스템이, 컴파일되어 상기 호스트 컴퓨터의 분산 네트워크 파일 시스템에 저장되어 있는 프로그램의 바이너리 코드를 읽어와서 실행시키는 이동 단말 장치.

청구항 8.

제7항에 있어서

상기 중앙 처리 장치는 상기 프로그램의 바이너리 코드 실행 결과를 상기 제2 인터페이스를 통하여 상기 콘솔 장치로 제공하는 이동 단말 장치.

청구항 9.

제7항 또는 제8항에 있어서

상기 제1 통신 인터페이스는 JTAG 통신 인터페이스이고, 상기 제2 통신 인터페이스는 시리얼 인터페이스이며, 상기 이동 단말 장치는 제1 및 제2 통신 인터페이스를 통하여 상기 호스트 컴퓨터 및 콘솔 장치와 피어-투-피어(peer-to-peer)로 연결되는 이동 단말 장치.

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 임베디드(embedded) 시스템에 관한 것으로, 더욱 상세하게 말하자면 임베디드 시스템의 소프트웨어 개발 및 실행 방법, 그리고 이를 이용한 이동 단말 장치에 관한 것이다.

일반적으로, 임베디드 시스템(Embedded System)이란 미리 정해진 특정 기능을 수행하기 위해 컴퓨터의 하드웨어와 소프트웨어가 조합된 전자 제어 시스템을 말하며, 필요에 따라서는 일부 기계가 포함될 수 있다. 즉, 우리 생활에서 쓰이는 각종 전자 기기, 가전제품, 제어장치 등은 단순히 회로로만 구성된 것이 아니라 마이크로컨트롤러(또는 마이크로 프로세서)가 내장되어 있고, 이 마이크로컨트롤러를 구동하여 특정한 기능을 수행하도록 프로그램이 내장되어 있는 시스템을 가리킨다. 쉽게 주위에서 볼 수 있는 임베디드 시스템으로는 예를 들면, DVD 플레이어, 셋탑박스, 이동 단말 장치, MP3 플레이어, 캠코더, 디지털 카메라, 냉장고, 세탁기와 같은 가전기기가 있다. 뿐만 아니라, 승강기, 항공기, 의료기기, 자동차, 공장제어, 통신 장비(교환기, 라우터 등)도 임베디드 시스템이다.

초기의 임베디드 시스템은 그 구성이 매우 단순하였으며, 4비트/8비트 컨트롤러에 제한된 동작을 하도록 하는 소프트웨어가 탑재된 시스템이 대부분이었다. 그러나, 요즘은 32비트 마이크로프로세서와 디지털 신호 처리기(DSP: Digital Signal Processor)가 일반적으로 사용됨에 따라 사용영역이 넓어지고 그에 따른 고성능의 소프트웨어도 함께 발달하게 되었다.

이러한 임베디드 시스템의 응용 분야가 다양해지면서, 예전에는 단순했던 임베디드 소프트웨어가 점점 더 복잡하고 다양한 기능을 처리하게 되었으며, 임베디드 시스템 제조업자들은 임베디드 소프트웨어를 빨리 개발하여 프로젝트 기간을 단축하는데 반드시 필요한 임베디드 소프트웨어 개발 환경을 강력히 원하고 있다.

임베디드 시스템은 대개 한정된 자원을 갖고 있기 때문에 임베디드 소프트웨어 개발 환경은 대개 호스트와 타겟으로 구성되며, 타겟에는 최소한의 개발 도구만 있고, 대부분의 도구는 자원이 풍부한 호스트에 존재한다. 임베디드 소프트웨어 개발 환경은 자원이 한정된 타겟에서 실행될 프로그램을 편리하고 빨리 개발할 수 있게 해야 한다.

일반적으로, 임베디드 시스템의 소프트웨어 개발은 크로스 툴체인(cross tool chain)을 이용하여 생성된 프로그램 바이너리 코드를 해당 임베디드 시스템의 플래시 메모리에 기록한 다음, 이후 상기 코드를 메인 메모리에 로딩시켜 실행한다.

그런데 이러한 경우에 임베디드 시스템의 소프트웨어 개발 단계에서 빈번하게 프로그램 수정이 일어나면, 소프트웨어 개발 주기가 연장되고 플래시 메모리의 수명을 단축시키는 문제가 발생한다. 즉, 프로그램이 수정이 발생될 때마다 임베디드 시스템의 플래시 메모리에 수정된 정보를 기록하는 과정이 수행됨으로써, 빈번한 정보 기록에 의하여 플래시 메모리의 수명이 단축된다.

발명이 이루고자 하는 기술적 과제

그러므로 본 발명이 이루고자 하는 기술적 과제는 보다 편리하고 효율성 있는 임베디드 시스템의 소프트웨어 개발 환경을 구축하고자 하는데 있다.

특히, 본 발명이 이루고자 하는 기술적 과제는 임베디드 시스템의 소프트웨어를 개발하는데 있어서 분산 네트워크 파일 시스템 기술을 이용하여, 임베디드 시스템의 플래시 메모리로의 불필요한 정보 기록 동작을 가능한 감소시키고자 하는데 있다.

발명의 구성

이러한 기술적 과제를 달성하기 위한 본 발명의 특징에 따른 임베디드 시스템의 소프트웨어 개발 및 실행 방법은, 로컬 네트워크를 통하여 임베디드 시스템과 호스트 컴퓨터, 콘솔 장치, 그리고 개발자 장치가 연결되어 있고, 상기 임베디드 시스템의 소프트웨어를 개발하고 실행시키는 방법에서, a) 상기 호스트 컴퓨터로부터 부트로더 파일, 커널 파일, 루트 파일 시스템 이미지 파일을 제공받아, 상기 임베디드 시스템의 플래시 메모리에 저장하는 단계; b) 상기 임베디드 시스템의 플레

시 메모리에 저장된 부트로더 파일 및 커널 파일의 실행에 따라 커널 자동 부팅이 수행되는 단계; c) 상기 커널 자동 부팅 수행에 따라 상기 플래시 메모리에 저장된 루트 파일 시스템이 임베디드 시스템의 루트에 마운트되는 단계; d) 상기 임베디드 시스템의 요청에 따라 상기 호스트 컴퓨터의 분산 네트워크 파일 시스템(이하, 제1 분산 네트워크 파일 시스템이라고 명명함)이 상기 루트 파일 시스템의 서브 디렉토리에 마운트되는 단계; e) 상기 임베디드 시스템에 마운트된 분산 네트워크 파일 시스템(이하, 제2 분산 네트워크 파일 시스템이라고 명명함)이 호스트 컴퓨터로부터 전달되는 바이너리 코드 실행 요청에 따라, 상기 호스트 컴퓨터에서 컴파일되어 상기 제1 분산 네트워크 파일 시스템에 저장되어 있는 프로그램의 바이너리 코드를 읽어오는 단계; 및 f) 상기 임베디드 시스템의 제2 분산 네트워크 파일 시스템이 상기 프로그램 바이너리 코드를 실행시키는 단계를 포함한다.

여기서, 상기 e) 단계는 상기 개발자 장치로부터 소정의 프로그램이 상기 호스트 컴퓨터로 제공되는 단계; 상기 호스트 컴퓨터가 상기 프로그램을 컴파일하고, 컴파일에 따라 생성된 프로그램 바이너리 코드를 제1 분산 네트워크 파일 시스템에 저장하는 단계; 상기 호스트 컴퓨터가 상기 임베디드 시스템으로 바이너리 코드 실행을 명령하는 단계; 상기 임베디드 시스템의 제2 분산 네트워크 파일 시스템이 상기 명령에 따라 상기 호스트 컴퓨터의 제1 분산 네트워크 파일 시스템에 저장된 바이너리 코드를 읽어오는 단계를 포함한다.

또한 본 발명의 다른 특징에 따른 이동 단말 장치는, 로컬 네트워크를 통하여 호스트 컴퓨터, 콘솔 장치, 그리고 개발자 장치가 연결되어 있는 이동 단말 장치에서, 상기 호스트 컴퓨터와의 데이터 송수신을 수행하는 제1 인터페이스; 상기 콘솔 장치와의 데이터 송수신을 수행하는 제2 인터페이스; 상기 호스트 컴퓨터로부터 제공받은 부트로더 파일, 커널 파일, 루트 파일 시스템 이미지 파일이 저장되어 있는 플래시 메모리; 상기 플래시 메모리에 저장된 각 파일들이 로딩되어 구동되는 메인 메모리; 상기 부트로더 파일 및 커널 파일의 구동에 따라 자동 커널 부팅이 이루어지면, 상기 플래시 메모리에 저장된 루트 파일 시스템을 루트에 마운트시키고, 상기 호스트 컴퓨터로부터 분산 네트워크 파일 시스템을 제공받아 상기 루트 파일 시스템의 서브 디렉토리에 마운트시키는 중앙 처리 장치를 포함하고, 상기 호스트 컴퓨터로부터 전달되는 바이너리 코드 실행 요청에 따라 상기 서브 디렉토리에 마운트된 분산 네트워크 파일 시스템이, 컴파일되어 상기 호스트 컴퓨터의 분산 네트워크 파일 시스템에 저장되어 있는 프로그램의 바이너리 코드를 읽어와서 실행시킨다.

이하, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 본 발명을 용이하게 실시할 수 있는 가장 바람직한 실시 예를 첨부된 도면을 참조로 하여 상세히 설명한다. 이하의 실시 예는 임베디드 시스템으로 임베디드 이동 단말 장치를 예로 하여 본 발명에 따른 임베디드 소프트웨어 개발 방법을 설명하지만, 본 발명에 따른 임베디드 시스템은 이동 단말 장치에 한정되지 않는다.

본 발명의 실시 예에서는 소프트웨어 개발 단계에서 수정된 프로그램 바이너리 코드를 이동 단말 장치의 플래시 메모리에 기록하지 않고 분산 네트워크 파일 시스템을 이용하여 이동 단말 장치에서 실행시킨다. 즉, 수정된 프로그램 바이너리 코드를 분산 네트워크 파일 시스템에 직접 반영한 다음에, 로컬 네트워크에 접속되고 분산 네트워크 파일 시스템이 마운트된 이동 단말 장치에서 프로그램 바이너리 코드를 실행할 수 있도록 한다. 이에 따라 이동 단말 장치의 소프트웨어 개발 완료 단계에서 최적화된 프로그램 바이너리 코드만을 이동 단말 장치의 플래시 메모리에 기록할 수 있으므로, 불필요한 플래시 메모리로의 코드 기록 횟수를 감소시킬 수 있다.

도 1은 본 발명의 실시 예에 따른 이동 단말 장치의 소프트웨어 개발을 위한 네트워크 구성을 나타낸 도이며, 도 2는 각 장치들의 구조도이다.

첨부한 도 1에 도시되어 있듯이, 본 발명의 실시 예에서 임베디드 이동 단말 장치의 소프트웨어 개발을 위하여, 이동 단말 장치(100), 호스트 컴퓨터(200), 콘솔 장치(300), 이동 단말 소프트웨어 개발자 장치(400)가 로컬 네트워크로 연결된다. 구체적으로, 이동 단말 장치(100)와 호스트 컴퓨터(200)는 JTAG(joint test action group) 통신 인터페이스를 통하여 피어-투-피어(peer-to-peer)로 연결되고, 이동 단말 장치(100)와 콘솔 장치(300)는 시리얼 통신 인터페이스(130)를 통하여 피어-투-피어로 연결된다.

여기서, 타겟 디바이스인 이동 단말 장치(100)는 도 2에 도시되어 있듯이, 중앙 처리 장치(110), 메인 메모리(120), 플래시 메모리(130), 시리얼 통신 인터페이스(140), JTAG 통신 인터페이스(150), 네트워크 통신 인터페이스(160), 사용자 인터페이스를 위한 입출력 장치 통신 인터페이스(170)를 포함하는 형태로 이루어진다. 플래시 메모리(130)는 부트로더 이미지, 커널 이미지, 루트 파일 시스템 이미지 저장을 위한 영역을 포함한다.

여기서 중앙 처리 장치(110)는 각 인터페이스를 통하여 호스트 컴퓨터(200) 및 콘솔 장치(300) 등과 통신하여, 개발자 장치(400)에서 개발되어 호스트 컴퓨터(200)에 저장되는 프로그램을 이동 단말 장치(100)에서 실행시키고, 그 결과를 콘솔 장치(300)를 통하여 개발자 장치(400)로 제공한다.

이러한 본 발명의 실시 예에 따른 이동 단말 장치(100)는 위에 기술된 구성 요소 이외에도 데이터 송수신 및 데이터 처리를 위한 다수의 수단을 더 포함할 수 있으며, 이러한 수단들은 이미 공지된 기술이므로 여기서는 상세한 설명을 생략한다.

또한 도 2에 도시되어 있듯이, 호스트 컴퓨터(200)도 타겟 디바이스와 동일한 하드웨어 환경을 구비하여, 중앙 처리 장치(210), 메인 메모리(220), 플래시 메모리(230), 하드디스크(240), 병렬 통신 인터페이스(250), 시리얼 통신 인터페이스(260), 네트워크 통신 인터페이스(270), 입출력 장치 통신 인터페이스(280)로 이루어진다. 호스트 컴퓨터(200)의 중앙 처리 장치(210)는 타겟 디바이스의 소프트웨어 개발을 진행하기 위한 크로스 컴파일 환경인 크로스 툴체인 컴파일러(211)를 포함한다. 크로스 툴체인 컴파일러(211)는 각종 소스들을 컴파일하고 빌드하여 실행 바이너리를 생성하는데 필요한 각종 유틸리티 및 라이브러리를 포함한다. 이동 단말 장치(100)는 JTAG 통신 인터페이스(150)를 통하여 호스트 컴퓨터(200)와 연결되어, 이동 단말 장치의 JTAG 통신 포트와 호스트 컴퓨터의 병렬 통신 포트가 JTAG 통신 인터페이스로 연결된다.

또한 콘솔 장치(300)도 중앙 처리 장치(310), 메인 메모리(320), 플래시 메모리(330), 하드디스크(340), 시리얼 통신 인터페이스(350), 네트워크 통신 인터페이스(360), 입출력 장치 통신 인터페이스(370)로 이루어진다.

다음에는 이러한 구조를 토대로 하여 본 발명의 실시 예에 따른 임베디드 이동 단말 장치의 소프트웨어 개발 환경을 구성하고 소프트웨어를 개발 및 실행시키는 방법에 대하여 설명한다.

임베디드 이동 단말 장치의 소프트웨어 개발 환경을 구성하기 위하여, 먼저, 이동 단말 장치(100), 호스트 컴퓨터(200), 콘솔 장치(300)의 하드웨어를 도 1에 도시된 바와 같이 구성하고, 이동 단말 소프트웨어 개발자 장치(400)의 하드웨어를 더 미 터미널 또는 호스트 컴퓨터 수준의 장치로 구성한다.

그리고, 이동 단말 장치(100)의 JTAG 통신 포트와 호스트 컴퓨터의 병렬 통신 포트를 JTAG 통신 인터페이스로 연결한다.

이와 같이 각 장치간의 네트워크 연결이 이루어진 후에, 다음과 같이 호스트 컴퓨터(200)로부터 이동 단말 장치(100)로의 이미지 파일 기록 과정이 수행된다.

도 3에 본 발명의 실시 예에 따라 임베디드 이동 단말 장치의 소프트웨어 개발 환경을 구성하는데 있어서, 이동 단말 이미지 파일이 기록되어 있는 상태의 이동 단말 장치의 구조가 개념적으로 도시되어 있다.

이동 단말 장치의 실행 환경을 구성하기 위하여, 이동 단말 장치(100)의 플래시 메모리(130)에 부트로더 이미지, 커널 이미지, 루트 파일 시스템 이미지 저장을 위한 영역을 구성한다. 도 3에 도시되어 있듯이, 이후 호스트 컴퓨터(200)로부터 생성되는 부트로더 파일, 커널 파일, 루트 파일 시스템 파일이 이미지 형태로 이동 단말 장치(100)의 플래시 메모리(130)의 해당 영역에 저장된다.

도 4는 본 발명의 실시 예에 따른 이동 단말 장치의 이미지 파일 기록 과정을 나타낸 흐름도이다.

첨부한 도 4에 도시되어 있듯이, 호스트 컴퓨터(200)는 크로스 툴체인 컴파일러(201)를 통하여 부트로더 파일을 컴파일시키고, 컴파일된 부트로더 파일을 JTAG 통신 인터페이스(140)를 통하여 이동 단말 장치(100)로 전달한다(S100~S120). 전달된 부트로더 파일은 이동 단말 장치(100)의 플래시 메모리(120)의 부트로더 이미지 영역에 이미지 형태로 기록된다. 이동 단말 장치(100)의 플래시 메모리(130)에 저장되는 부트로더 이미지(131)는 통신 인터페이스 초기화(시리얼 통신, 네트워크 통신) 기능과 부트로더 프롬프트 처리 기능으로 구성된 프로그램이다.

이러한 부트로더 파일이 이미지 형태로 기록된 다음, 이동 단말 장치(100)의 전원을 차단시킨 다음에 다시 재공급한다(S130). 이러한 전원 공급 차단 및 재공급 과정은 개발자에 의하여 수행되거나 호스트 컴퓨터(200)의 제어에 따라 이루어질 수도 있다.

이동 단말 장치(100)로 전원이 재공급되면 플래시 메모리(130)에 저장되어 있던 부트로더 이미지가 메인 메모리(120)로 로딩되면서 실행된다. 그리고 호스트 컴퓨터(200)의 크로스 툴체인 컴파일러(211)에 의해 커널 파일이 생성된다.

이후 콘솔 장치(300)와의 통신 인터페이스 설정 경로를 통하여 콘솔 장치(300)로부터 부트로더 명령어 모드가 선택된다(S140). 이에 따라 이동 단말 장치(100)의 메인 메모리(120)에서 실행 중인 부트로더(131)에 의하여 콘솔 장치(300)의 화면에 부트로더 명령어 프롬프트가 출력되고(S150), 콘솔 장치(300)로부터 생성되고 커널 이미지 전송을 위한 부트로더 명령어가 통신 인터페이스를 통하여 이동 단말 장치(100)로 전달된다(S160).

이동 단말 장치(100)의 부트로더(131)는 콘솔 장치(300)로부터 커널 이미지 전송을 위한 명령어가 전송되면, 로컬 네트워크 통신 인터페이스(160)를 통하여 호스트 컴퓨터(200)로 커널 이미지 전송을 요청한다(S170). 이에 따라, 호스트 컴퓨터(200)는 크로스 툴체인 컴파일러(211)에 의해 컴파일된 이동 단말 장치의 커널 파일을 로컬 네트워크(500)를 통하여 이동 단말 장치(100)로 전송한다(S180). 이동 단말 장치(100)는 커널 파일을 전송받아 플래시 메모리(130)의 커널 이미지영역(132)에 이미지 형태로 저장한다(S190).

다음, 콘솔 장치(300)에 대해 부트로더 명령어 모드가 다시 선택되고(S200), 이동 단말 장치(100)의 메인 메모리(110)에서 실행 중인 부트로더(131)에 의하여 콘솔 장치(300)의 화면 상에 부트로더 명령어 프롬프트가 출력됨에 따라 입력되는 파일 시스템 이미지 전송을 위한 부트로더 명령어가 이동 단말 장치(100)로 전달된다(S210~S220). 이 전달되는 부트로더 명령어에 따라 이동 단말 장치(100)의 부트로더(131)는 로컬 네트워크 통신 인터페이스(160)를 통하여 호스트 컴퓨터(200)로 루트 파일 시스템 이미지 전송을 요청한다(S230). 호스트 컴퓨터(200)는 크로스 툴체인 컴파일러(211)에 의해 생성된 각종 라이브러리 및 환경 파일들로 구성되는 루트 파일 시스템 파일(또는 호스트 컴퓨터(200)의 공개된 저널링(Journaling) 파일 시스템 제작 프로그램을 이용하여 루트 파일 시스템 파일을 생성할 수도 있다.)을 로컬 네트워크를 통하여 이동 단말 장치(100)로 전송한다(S240). 전송된 루트 파일 시스템 파일은 이동 단말 장치(100)의 플래시 메모리(130)의 루트 파일 시스템 이미지 영역(133)에 이미지 형태로 기록된다(S250).

위에 기술된 바와 같이, 호스트 컴퓨터(200)에 의하여 컴파일된 부트로더 파일, 커널 파일, 루트 파일 시스템 파일이 이미지 형태로 이동 단말 장치(100)의 플래시 메모리(130)에 기록되면, 다음과 같이 이동 단말 장치(100)의 소프트웨어 개발 및 실행이 수행된다.

도 5는 본 발명의 실시 예에 따른 이동 단말 장치의 소프트웨어 개발 및 실행 과정을 나타낸 흐름도이다.

첨부한 도 5에 도시되어 있듯이, 이동 단말 장치(100)와 콘솔 장치(300)가 시리얼 통신 인터페이스를 통하여 연결되고(S300), 이동 단말 장치(100)와 호스트 컴퓨터(220) 및, 호스트 컴퓨터(200)와 이동 단말 소프트웨어 개발자 장치(400)가 로컬 네트워크로 각각 연결된 상태에서(S310), 이동 단말 장치(100)의 커널 부팅 및 파일 시스템 마운트 절차를 수행한다.

이를 위하여, 이동 단말 장치(100)의 전원을 차단한 후 재공급 하여 이동 단말 장치(100)를 커널 자동 부팅 모드로 동작시킨다(S320~S330). 즉, 플래시 메모리(130)에 저장되어 있던 부트로더 이미지가 메인 메모리(120)에 로딩되면서 실행되고, 부트로더가 이동 단말 장치의 하드웨어를 초기화시키고, 플래시 메모리(130)에 저장된 커널 이미지(132)를 메인 메모리(120)로 로딩시킨 다음, 실행 권한을 커널로 인계한다. 이후, 커널의 제어에 따라 호스트 컴퓨터(200)와 통신하여 bootp 클라이언트 절차 및 tftp 클라이언트 절차가 이루어진다. 그리고 커널(132)의 제어에 따라 플래시 메모리(130)에 저장되어 있던 파일 시스템(133)이 루트에 마운트된다(S340).

이러한 이동 단말 장치(100)의 커널 자동 부팅 과정 중에 호스트 컴퓨터(200)는 네트워크 파일 시스템이 이동 단말 장치(100)의 루트 파일 시스템(205)의 서브 디렉토리에 마운트되도록 파일 시스템 테이블 환경 파일을 구성한다. 이후 이동 단말 장치(100)로부터의 네트워크 파일 시스템 마운트 요청에 호스트 컴퓨터(200)가 응답하여, 이동 단말의 소프트웨어 개발 단계에서 빈번하게 변경이 이루어지는 파일들로 구성된 분산 네트워크 파일 시스템을 로컬 네트워크(500)를 통하여 이동 단말 장치(100)의 루트파일 시스템(133)의 서브 디렉토리에 마운트시킨다(S350~S360).

이후 개발자는 이동 단말 소프트웨어 개발자 장치(400)를 통하여 호스트 컴퓨터(200)에 원격으로 접속하여, 호스트 컴퓨터(200)에서 컴파일된 파일을 네트워크 파일 시스템(CNF1) 영역에 복사시켜, 상기 파일이 이동 단말 장치(100)의 네트워크 파일 시스템(CNF2)에 반영되도록 한다.

보다 구체적으로, 이동 단말 소프트웨어 개발자 장치(400)에서 로컬 통신 네트워크 인터페이스를 통하여 호스트 컴퓨터(200)에 원격 접근하여, 개발된 이동 단말 프로그램을 호스트 컴퓨터(200)로 제공한다. 이동 단말 소프트웨어 개발자 장치(400)의 원격 제어에 따라 호스트 컴퓨터(200)는 크로스 툴체인 컴파일러(211)를 통하여 이동 단말 프로그램을 컴파일하고, 컴파일에 따라 생성된 바이너리 코드 파일을 네트워크 파일 시스템(CNF1) 영역에 복사한다(S370).

다음 이동 단말 소프트웨어 개발자 장치(400)는 로컬 통신 네트워크 인터페이스(160)를 통하여 콘솔 장치(300)에 원격 접근하여 프로그램 바이너리 코드 실행을 요청한다(S380). 이에 따라 콘솔 장치(300)는 시리얼 통신 인터페이스(350)를 통하여 이동 단말 장치(100)에 프로그램 바이너리 코드 실행을 지시한다(S390). 콘솔 장치(300)로부터 제공되는 코드 실행 지시에 따라 이동 단말 장치(100)의 중앙 처리 장치(110)는 플래시 메모리(130)의 루트 파일 시스템의 서브 디렉토리에

마운트된 네트워크 파일 시스템(CNF2)을 메인 메모리(120)로 로딩시켜 실행시키고, 실행에 따라 네트워크 파일 시스템(CNF2)은 호스트 컴퓨터(200)의 네트워크 파일 시스템(CNF1)에 저장되어 있는 바이너리 코드 파일을 읽어와서 실행시킨다.

이후, 이동 단말 장치(100)의 중앙 처리 장치(110)는 상기 바이너리 코드 파일의 실행 결과를 콘솔 장치(300)로 전달하고(S400), 전달된 실행 결과는 콘솔 장치(300)의 네트워크 통신 인터페이스(360)를 통하여 이동 단말 소프트웨어 개발자 장치(400)로 전달된다. 따라서, 소프트웨어 개발자는 콘솔 장치(300)를 통하여 전달되는 실행 결과를 통하여 새로이 개발된 소프트웨어가 이동 단말 장치(100)에서 어떻게 수행되었는지를 확인하게 된다(S410).

위에 기술된 바와 같은 방법에 따라 개발자는 용이하게 개발된 소프트웨어를 이동 단말 장치(100)에서 실행시킨 결과를 확인할 수 있으며, 이후, 최종적으로 최적의 소프트웨어가 개발 완료되면, 콘솔 장치(300) 등을 통하여 해당 소프트웨어를 이동 단말 장치(100)의 플래시 메모리(130)에 저장하여 사용될 수 있다.

위에 기술된 방법은 컴퓨터가 읽을 수 있는 기록 매체에 저장되는 프로그램 형태로 구현될 수 있다. 기록 매체로는 컴퓨터에 의하여 읽혀질 수 있는 데이터가 저장되는 모든 종류의 기록 장치가 포함될 수 있으며, 예를 들어, CD-ROM, 자기 테이프, 플로피 디스크 등이 있으며, 또한 캐리어 웨이브(예컨대 인터넷을 통한 전송)의 형태로 구현되는 것도 포함된다.

비록, 본 발명이 가장 실제적이며 바람직한 실시 예를 참조하여 설명되었지만, 본 발명은 상기 개시된 실시 예에 한정되지 않으며, 후술되는 특허청구범위 내에 속하는 다양한 변형 및 등가물들도 포함한다.

발명의 효과

이상 위에 기술된 바와 같은 본 발명에 따르면, 분산 네트워크 파일 시스템 기술을 이용하여 소프트웨어 개발 단계에서 수정된 프로그램 바이너리 코드를 이동 단말 즉, 임베디드 시스템의 플래시 메모리에 기록하지 않고 분산 네트워크 파일 시스템에 반영하여, 이후 로컬 네트워크를 통하여 접속된 임베디드 시스템에서 프로그램 바이너리 코드를 실행할 수 있도록 함으로써, 임베디드 시스템 소프트웨어 개발 완료 단계에서 최적화된 프로그램 바이너리 코드 파일만을 임베디드 시스템의 플래시 메모리에 기록할 수 있다.

그에 따라, 소프트웨어 개발시마다 임베디드 시스템의 플래시 메모리에 기록하는 횟수를 감소시킬 수 있다. 또한 플래시 메모리로의 불필요한 정보 기록 횟수의 감소에 따라 플래시 메모리의 수명을 보다 연장시킬 수 있다.

또한 변경된 이동 단말 장치의 소프트웨어를 플래시 메모리에 기록하지 않고도 즉각 분산 네트워크 파일시스템에 반영하여 임베디드 시스템에서 실행해 볼 수 있다.

또한 네트워크를 통하여 임베디드 시스템의 소프트웨어를 보다 용이하게 공동으로 개발할 수 있다.

도면의 간단한 설명

도 1은 본 발명의 실시 예에 따른 임베디드 시스템의 소프트웨어 개발을 위한 장치들의 네트워크 연결도이다.

도 2는 도 1에 도시된 각 장치들의 구조도이다.

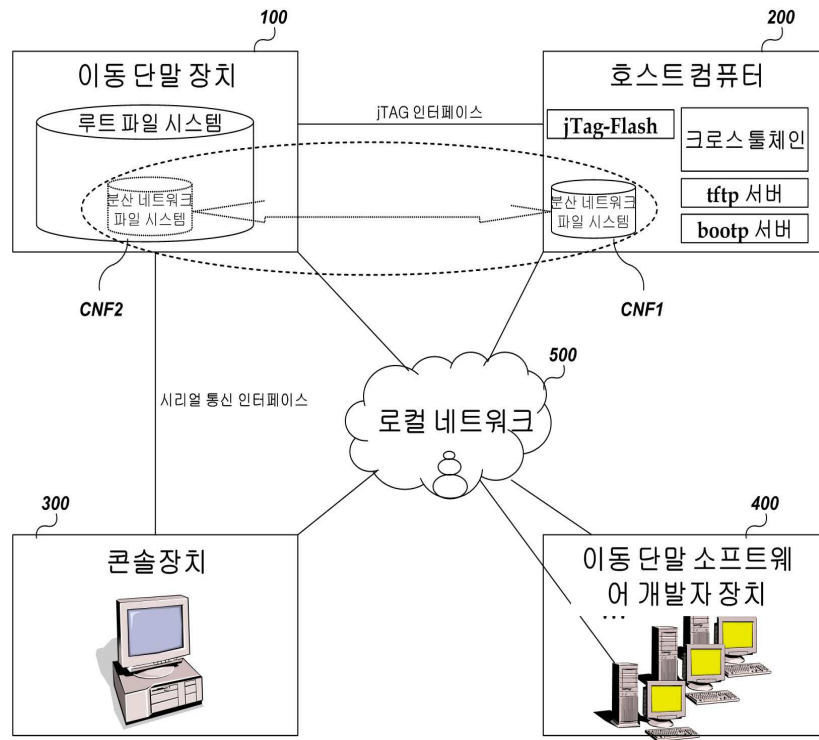
도 3은 본 발명의 실시 예에 따라 임베디드 이동 단말 장치의 소프트웨어 개발 환경을 구성하는데 있어서, 이동 단말 이미지 파일이 기록되어 있는 상태의 이동 단말 장치의 구조를 개념적으로 나타낸 도이다.

도 4는 본 발명의 실시 예에 따른 이동 단말 장치의 이미지 파일 기록 과정을 나타낸 흐름도이다.

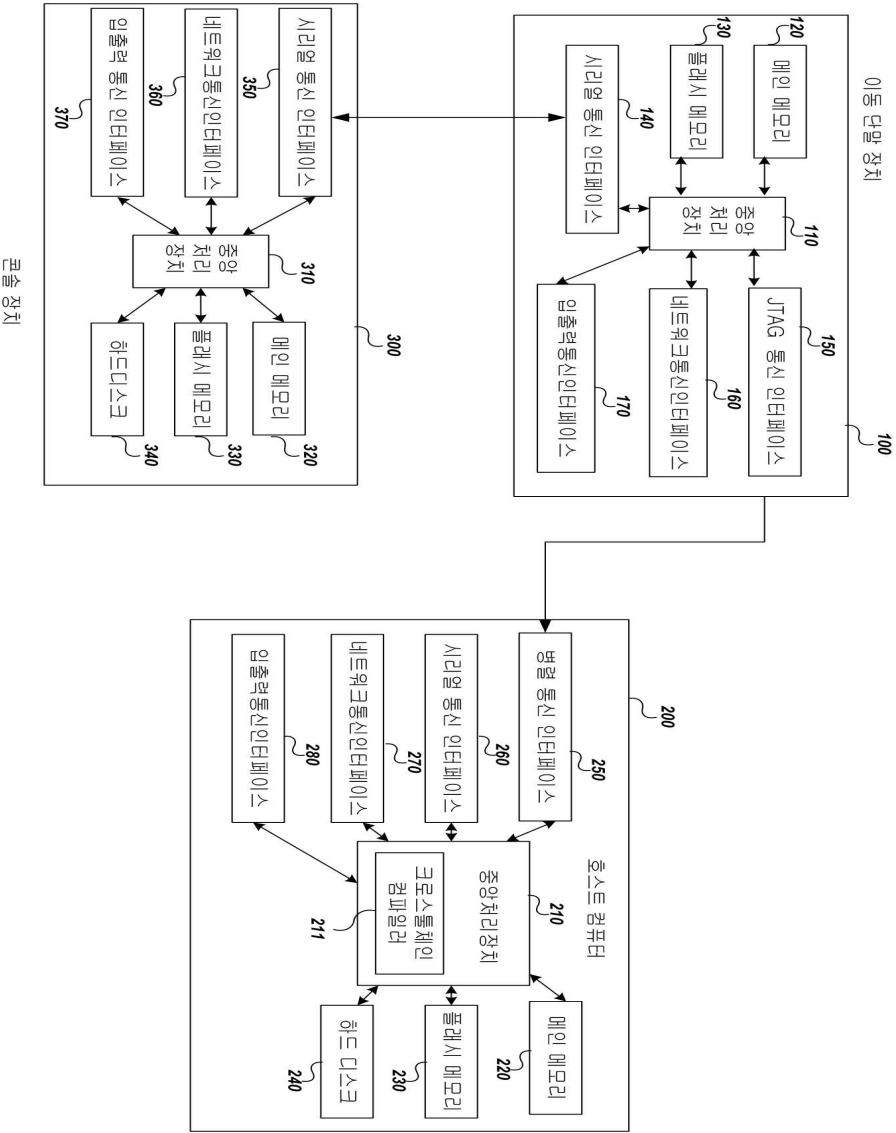
도 5는 본 발명의 실시 예에 따른 이동 단말 장치의 소프트웨어 개발 및 실행 과정을 나타낸 흐름도이다.

도면

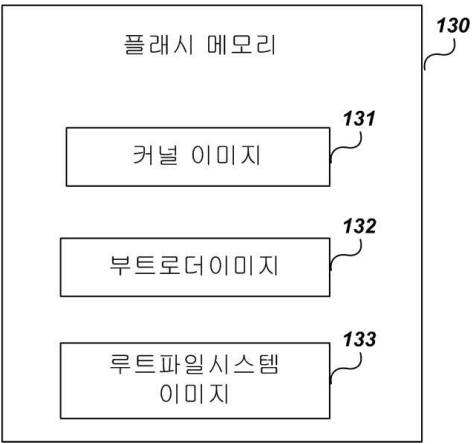
도면1



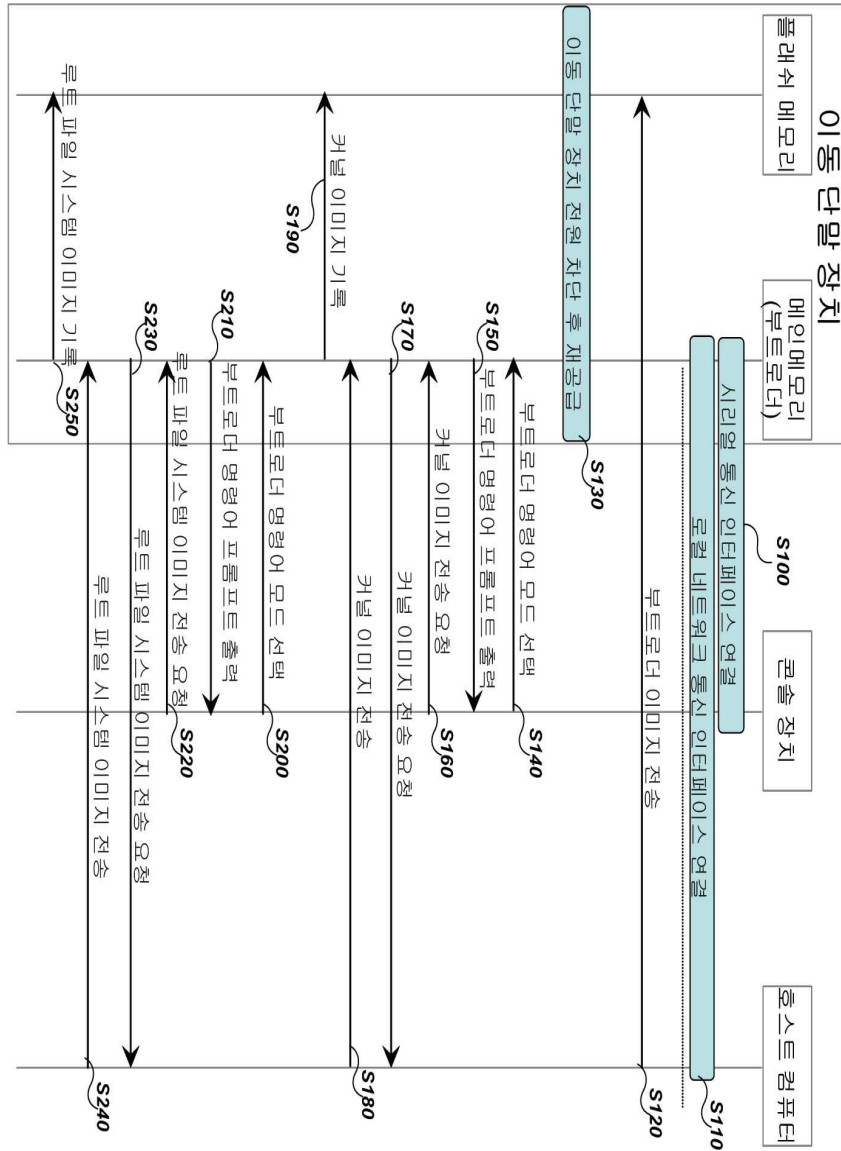
도면2



도면3



도면4



도면5

