논문 2012-07-14

휴대용 게임기를 이용한 공개 SW 기반 임베디드 소프트웨어 교육 IDE 환경 구성

(Implementation of Open Source Embedded Software Educational Integrated Development Environment Using Portable Game Console)

이 민 석*
(Minsuk Lee)

Abstract: : In this paper we describe the integrated development environment for embedded software education. Our environment uses a cheap and widely available portable game console, Nintendo DS-lite. We have integrated open source development tools, and have implemented download shell, remote debugger. The results are all open source, and were used for university class for real embedded software education.

Keywords : Embedded Software Education, Game Console, IDE, Open Source

Ⅰ. 서론

임베디드 소프트웨어는 IT 뿐만 아니라 전 융합 산업에 걸쳐 활용되는 기술 분야로서 전문 인력에 대한 수요가 지속적으로 증가하고 있다. 임베디드 시스템 또는 임베디드 소프트웨어 교육은 일반적으로 임베디드 하드웨어 보드와 그와 연관된 소프트웨어 개발 도구를 이용한다. 현재 많은 대학에서는 그림 1의 예에서 보는 것과 같이 여러 타입의 ARM CPU를 기반으로 하는 하드웨어 [1]와 임베디드 리눅스 그리고 안드로이드 플랫폼을 중심으로 교육이 이루어지고 있다.

위와 같은 장비를 이용하여 행해지고 있는 임베디드 시스템 교육 환경은 다음과 같은 단점을 가지고 있다. 첫째, 임베디드 시스템 개발 장비는 세부 사양에 따라 적어도 백만 원에서 수백만 원에 이르기 때문에 교육 기관에 상당한 비용 부담을 주며, 개인 개발자가 구입하기에는 가격이 매우 비싸다. 둘째, 기존 장비는 그림에서 보는 것처럼, 스마트폰, 게임기 등 학생들이 실생활에서 보고 있는 임베디드 시스템과는 전혀 다른 모양으로서, 임베디드 시스템을 처음 배우는 학생들에게 친근하지 않아 현실감이 부족하여 동기 유발이 어렵다. 셋째, 효과적인 교육을 위한 디버깅 환경의 구축을 위해서는 고가의 디버깅 장비가 별도로 필요하다.

또한, 학생들의 교과목 이수 체계, 교과목의 교육 목표와 이수 수준에 따라 차이가 있겠지만, 전형적인 임베디드 소프트웨어 교육 항목은 임베디드 리눅스를 위한 디바이스 드라이버 작성, 커널 모듈의 설계 구현, 안드로이드 응용 프로그램과 앞서 만

그림 1. 임베디드 시스템 교육 장비의 예
Fig. 1. Typical Hardware for Embedded System Education

* 교신저자(Corresponding Author)
논문접수 : 2012. 01. 19., 수정일: 2012. 02. 07.,
채택확정 : 2012. 02. 15.
이민석 : 한성대학교 컴퓨터공학과
※ 본 연구는 한성대학교 교내연구비의 지원으로 이루어졌음.

ISSN : 1975-5066

들어진 디바이스 드라이버나 모듈 등 새로 추가된 커널 기능과의 연계 기능 구현 등이 포함된다. 이러한 교육 항목은 운영체제 기초 이론을 이수하고, 성숙되지 않은 시스템 프로그램 능력을 가진 학생들에게는 비교적 힘든 주제로 수업에 참가하는 대부분의 학생들이 과목에서 제시된 교과목 학습 성과를 달성하지 못한다.

본 연구에서는 적은 비용으로 높은 교육 효과를 얻을 수 있는 임베디드 소프트웨어 교육을 위한 환경 구성 방안을 기술하고자 한다. 제안하는 교육 환경은 저렴한 휴대용 상용 게임기를 이용하여 피교육자들에게 흥미와 성취감을 불러 일으켜서 교육을 성과를 높일 수 있는 기대효과를 가지고 있다.

연구에서 구성된 교육 환경에서는 가능한 모든 부분에 공개 소프트웨어 개발 도구를 활용하였다. 임베디드 시스템 개발에 있어서, 통합 개발 환경은 소프트웨어 개발의 생산성과 코드의 품질 개선을 위해 매우 중요하다 [2, 3]. 국내에서는 ETRI에서 Qplus 및 나노 Qplus를 위한 Esto 통합개발 환경을 이클립스 [4] 기반으로 개발한 전례가 있다 [5]. 이클립스는 가장 성공적인 공개 소스 개발 도구 기반으로 현재 가장 많이 사용되고 있다.

본 과제에서도 이클립스 통합 개발 환경 상에서 프로젝트 수준의 관리, 버전 관리, 컴파일, USB를 이용한 다운로드 및 원격 디버깅이 가능한 도구를 개발하였다. 또한 임베디드 소프트웨어 교육을 위한 기반으로 역시 공개 소프트웨어인 실시간 운영체제를 이식하고, 그 위에서 교육을 위한 모든 실습 프로그램 예제의 실행이 가능하도록 하여, 임베디드 소프트웨어 교육에 반드시 필요한 여러 운영체제 이슈들을 경험할 수 있도록 구성하였다.

본 논문은 II 장에서 연구 개발의 범위로 전체적인 환경 구성과 이 구성을 위해 활용된 공개 소프트웨어들과, 추가로 구현된 소프트웨어 및 하드웨어의 내역을 소개하고, III 장에서는 구현된 소프트웨어 도구들을 구체적으로 기술하고, Ⅳ장에서 결론과 발전 방향에 대한 제시를 한다.

II. 연구 범위

이 장에서는 임베디드 소프트웨어 교육을 위한 실습 환경이 갖추어야 할 여러 가지 요소들을 살펴보고, 연구에서 구현된 교육 환경에 대하여 기술하고자 한다. 연구에서는 학생들이 리눅스나 윈도우즈 PC를 이용한 개발 호스트에서 실행되는 통합 개발 환경에서 실시간 운영체제를 이용한 임베디드 소프트웨어를 개발하여, 상용 게임기의 게임팩과 같은 형태로 빌드한 뒤, 게임기로 프로그램 다운로드 및 실행, 원격 디버깅이 가능하게 하는 일련의 개발 도구 및 환경을 구현하였다.

성공적인 임베디드 소프트웨어 교육을 위해 고려가 필요한 소프트웨어 요소들은 다음과 같다.

- 교차 개발 환경 : 개발 호스트와 타겟 시스템이 같지 않은 환경에서, 호스트에서 모든 개발 과정을 마치고, 타겟 시스템에 개발된 소프트웨어(바이너리)를 다운로드하고, 원격으로 디버깅을 할 수 있는 환경. 교차 개발 환경에는 호스트 측에서 실행되는 IDE (통합 개발 환경 - 프로젝트 관리 도구, 컴파일러, 어셈블러, 링커, 디버거), 다운로더, 타겟 시뮬레이터 등이 포함된다.
- 타겟 시스템의 환경 : 타겟 시스템은 개발 과정에서 호스트 컴퓨터에서 실행되는 통합 개발 환경의 동작에 대응하는 모듈들과, 응용 프로그램의 실행의 기반이 되는 운영체제 및 라이브러리를 포함하여야 한다. 여기에는 실행 바이너리를 수신하고, 수신된 바이너리들을 관리하고 실행하기 위한 쉘, IDE의 디버깅 기능에 대응하여 타겟 시스템에 다운로드된 소프트웨어를 제어하기 위한 디버그 스텁 모듈, 실시간 운영체제, 이 운영체제 위에서 여러 편의 기능을 제공하기 위한 통신, 파일 시스템 등의 라이브러리 등이 속한다.

본 연구에서 타겟 시스템으로 사용한 상용 게임기로는 국내에서 가장 높은 시장 점유율을 가진 휴대용 게임기인 닌텐도사의 닌텐도 DS-lite이다. 연구에서 구성된 전체 임베디드 소프트웨어 교육 환경은 그림 2와 같다.

교육 환경에 사용된 닌텐도 DS-lite는 국내뿐만

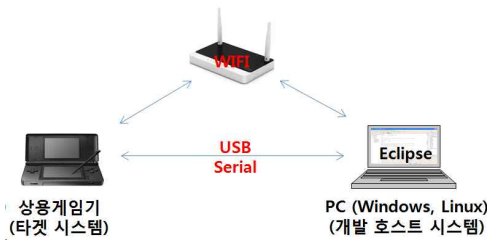

그림 2. 임베디드 소프트웨어 교육 환경
Fig. 2. Education Environment for Embedded Software

표 1. 닌텐도 DS-lite의 주요 하드웨어 사양
Table 1. Key Hardware Specification of Nintendo DS-lite

분류	사양
CPU	ARM946E-S 67MHz
	ARM7TDMI 33MHz
LCD	위: 18bit, 256x192
	아래: 18bit, 256x192 with Touch
ROM	256K Byte Flash
RAM	4M Byte SRAM (확장 가능)
통신	무선랜 : 802.11b (WEP)
그래픽	2D, 3D 가속 엔진
기타	키패드 (조이스틱), 마이크, Audio, Dual Slot, 1000mAh 배터리

아니라 세계에서 가장 많이 팔린 휴대용 게임기 가운데 하나로, 지금도 시중에서 10만원 내외의 가격으로 구입이 가능하다. 이 게임기는 ARM9, ARM7 등 두 개의 32비트 CPU, 두 개의 LCD 화면, 그래픽 가속 장치, 버튼, 터치스크린, 사운드 입출력, 무선랜 등 기존 임베디드 시스템 교육 장비가 필요로 하는 대부분의 장치들을 가지고 있다. 또, 소프트웨어 호환성이 보장되는 후속 버전(닌텐도 DS-i, 닌텐도 3DS)에는 듀얼 카메라와 외장 SD 카드 등 추가적인 입출력 장치도 내장되어 있다. 표 1은 닌텐도 DS-lite의 하드웨어 사양이다.

닌텐도 사에서 나온 여러 휴대용 게임기들은 게임 소프트웨어의 호환성 때문에, 하드웨어 적으로 매우 유사하다. 공식적으로는 하드웨어 사양을 공개하지 않고 있으나, 개인 게임 개발자 커뮤니티 등에서 역공학적으로 분석하여 구체적인 하드웨어 세부 사양과 동작 방식이 알려져 있고 [6] 그를 위한 라이브러리들도 많이 공개되어 있다. 본 연구에서는 이 자료를 바탕으로 타겟 시스템에 실시간 운영체제를 이식하고, 원격 디버깅 환경을 구축하는 것이 가능하였다.

임베디드 소프트웨어 개발 환경의 소프트웨어 구성은 그림 3과 같다. 그림에서 볼 수 있는 바와 같이 구성에 포함된 소프트웨어 가운데 가능한 모든 부분에 기존 공개 소스 소프트웨어를 활용하였다.

타겟 시스템인 닌텐도 DS-lite에는 실시간 운영체제로 공개 소스인 FreeRTOS [7]를 이식하여 사용하였으며, 원격 디버깅을 위하여 GDB [8, 9]

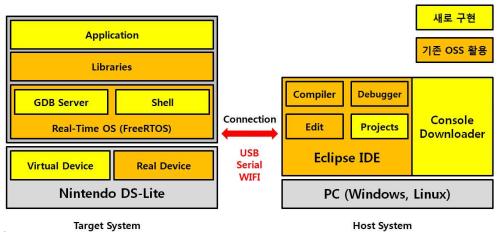

그림 3. 임베디드 소프트웨어 교육 환경 세부 구성
Fig. 3. System Architecture of Embedded Software Education Environment

서버(스텁 모듈)를 구현하였다. 그리고 호스트 시스템에서 개발된 프로그램을 다운로드 받고, 관리하기 위한 쉘 프로그램을 새로 구현하였다. 또 임베디드 소프트웨어 교육에 활용하기 위하여 실제 닌텐도 DS-lite가 가지고 있지 않은 장치들을 가상적으로 시뮬레이트하는 가상 디바이스를 추가하였다. 타겟 시스템에서 실행되는 프로그램과 링크되는 라이브러리는 devkitPro [10], Palib [11]를 수정하여 사용하였다.

호스트 시스템은 리눅스 운영체제나 윈도우즈 운영체제를 모두 활용할 수 있도록 하였다. 호스트의 통합 개발 환경으로는 이클립스 CDT를 이용하며, 컴파일러는 devkitPro [10]를 사용한다. 이클립스 CDT는 devkitPro가 제공하는 GNU C 컴파일러, 어셈블러, 링커, 디버거를 그대로 이용한다. 컴파일러와 바이너리 유틸리티는 워낙 리눅스 용으로 개발된 것이기 때문에, 윈도우즈에서는 cygwin과 같은 플랫폼 적응 라이브러리를 사용한다. 이클립스는 Java 개발에 주로 사용되는 IDE로서 많은 학생들이 이미 익숙해져 있기 때문에, 별도로 사용 교육이 필요 없는 도구이다. 버전 관리도구로는 SVN을 적용하였다. 개발된 실행 파일을 타겟 시스템에 USB 또는 무선랜을 통하여 다운로드하기 위한 콘솔 다운로더를 구현하였다. 모든 개발은 이클립스 IDE에서 이루어지며, 빌드된 바이너리 이미지는 다운로더를 통해 타겟 시스템인 닌텐도 DS-lite로 전송되어 실행된다. USB 연결을 통해 원격 디버깅 모드를 이용할 수도 있다. 또 호스트 시스템에서 닌텐도 DS 시뮬레이터를 이용하여 실행하는 것도 가능하다.

그림 3의 세부 구성에서 새로 구현한 모든 소프트웨어는 그 소스가 [12]에 공개되어 누구나 다운로드할 수 있도록 하였다.

그림 4. 확장 하드웨어 어댑터
Fig. 4. The Extension Adapter

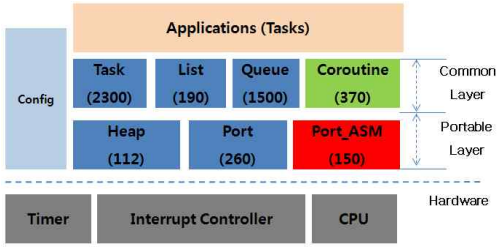

그림 5. FreeRTOS의 구조
Fig. 5. The Architecture of FreeRTOS

연구에서는 호스트와의 연결 방법이 무선랜 밖에 없는 닌텐도 DS-lite에 USB 또는 시리얼 포트 연결성을 부여하고, 가속 센서와 같은 추가적인 장치의 제공, 외부 장치 제어가 가능하도록 그림 4의 어댑터 하드웨어를 개발하였다. 무선랜은 개인 개발자들에게는 매우 유용하나, 보안이 적용된 환경에서는 여러 제약이 있기 때문에 사용이 편리한 USB 연결을 지원하였다.

이 어댑터는 닌텐도 DS-lite 입장에서는 게임팩처럼 동작한다. 어댑터는 독자적인 32-bit CPU를 가지고 있으며 기본적으로는 호스트 시스템과 닌텐도 DS-lite 사이의 통신을 중계한다. 호스트 시스템과 어댑터는 USB 또는 시리얼 인터페이스로 연결되며, 어댑터와 닌텐도 DS-lite 사이는 닌텐도의 게임팩 슬롯이 제공하는 SPI 인터페이스를 이용한 통신이 이루어진다. 이 어댑터에도 우리 실험 환경과 같은 실시간 운영체제인 FreeRTOS를 사용한다. 이 FreeRTOS는 CPU 제조사에서 이식하여 제공한 버전을 그대로 사용한다.

연구에서는 이 하드웨어 어댑터의 회로도 역시 [12]에 공개하고 있으며, 이 하드웨어의 장치들을 이용하기 위한 API를 만들어 두고 있다.

Ⅲ. 구현

이 장에서는 임베디드 소프트웨어 교육 환경을 구성하고 있는 주요 모듈들의 구체적인 구현 내용을 기술한다.

1. FreeRTOS

FreeRTOS [7]는 Richard Berry에 의해 설계된 작고 사용하기 쉽도록 디자인되어 있는 공개 소스 실시간 운영체제 (Real-Time Operating System)이다.

본 연구에서는 FreeRTOS를 타겟 시스템인 닌텐도 DS-lite에 이식하고 그 위에서 실시간 운영체제를 이용한 실습 교육 환경을 제공한다.

그림 5는 FreeRTOS의 구성을 나타낸다. 그림에서 괄호안의 숫자는 각 모듈의 라인 수이며, FreeRTOS가 수 천 라인의 작은 운영체제임을 바로 알 수 있다. FreeRTOS는 플랫폼에 상관없이 공통적으로 사용되는 Common Layer와 플랫폼 종속적인 Portable Layer로 소스코드가 구분되어 있다. FreeRTOS를 이식하기 위해서는 Portable Layer에 속하는 소스 코드를 수정해 주어야 한다.

FreeRTOS의 이식은 닌텐도 DS-lite에서 사용하고 있는 ARM9 프로세서 [13]의 레지스터와 중첩 인터럽트 모델(Nested Interrupt Model)을 적용하기 위한 FreeRTOS의 변수 등 문맥(Context)을 구성하고, 문맥전환(Context Switching) 기능을 하는 함수의 구현, 닌텐도 DS-lite의 타이머 장치를 이용하여 타이머 인터럽트 등록을 통해 이루어진다. 이 과정은 닌텐도 DS-lite와 ARM9 프로세서에 맞도록 플랫폼 종속적인 데이터 구조의 수정을 포함한다.

닌텐도 DS-lite는 독자적인 BIOS를 펌웨어 형태로 롬(ROM)에 가지고 있어 모든 인터럽트 벡터가 롬에 고정되어 있다. 따라서 모든 인터럽트가 BIOS의 인터럽트 서비스 루틴을 거쳐야 하기 때문에, FreeRTOS의 운영에 필요한 타이머 인터럽트 역시 두 단계의 인터럽트 서비스 과정을 거치므로 약간의 성능 저하를 일으키지만, 임베디드 소프트웨어 교육에는 전혀 문제가 없는 수준이다.

2. 실제 하드웨어 장치들에 대한 접근

닌텐도 DS-lite는 표1에서와 같이 ARM7, ARM9 2개의 프로세서를 사용하고 있다. 또한

2D,3D 그래픽 엔진, 2개의 256x192 해상도의 LCD, 12개의 버튼(상하좌우의 방향키, A, B, X, Y, L, R, Start, Select 버튼), 하단 화면의 터치스크린, 마이크와 스피커를 포함하는 16채널 사운드 장치, 무선 랜 등 다양한 장치들로 구성되어 있다. 이러한 장치들은 개발 환경에 포함되는 닌텐도 라이브러리 또는 I/O 맵핑 정보를 이용하여 직접적인 제어가 가능하다.

실제 입출력 장치의 일부(각종 스위치 입력, 터치 좌표 등)는 닌텐도 DS-lite의 ARM7 CPU에 의해 구동되며, ARM7과 ARM9이 공유하고 있는 메모리에 입출력 장치의 현재 상태가 매핑된다. 그 값은 실시간 운영체제가 운영되어 실습용 프로그램이 실행되고 있는 ARM9 CPU에서 참조된다. 실험 환경에서는 이 공유 메모리를 직접 참조하는 매크로를 제공함으로써 각종 입출력 장치를 실시간 운영체제의 응용 프로그램이 직접 이용할 수 있도록 하였다.

3. 가상 하드웨어 장치의 구성

가상 하드웨어 장치는 닌텐도 DS-lite에 있는 LCD 화면과 터치스크린 상에서 동작하며, 소프트웨어로 동작하는 시뮬레이터에 의해 구현된다.

연구에서는 우리 임베디드 소프트웨어 교육 환경에서 사용하는 닌텐도 DS-lite가 다양한 하드웨어 요소들을 이미 가지고 있으나, 실제 임베디드 시스템 하드웨어들이 많이 사용하는 비교적 간단한 하드웨어 장치들을 가상적으로 추가 제공함으로써 교육의 효과를 높이고자 하였다. 제공되는 가상 디바이스는 그림 6에서 보는 바와 같이, 2 개의 바(Bar) LED, 8 개의 7 세그먼트 LED, 채터링이 있는 스위치, 스캔 방식의 스위치 등이다. 그림 6의 위쪽 LCD에는 두 개의 바 LED와 8개의 7 세그먼트 LED가 있으며, 터치가 동작하는 아래쪽 LCD는 두 개의 가상 스위치, 두 개의 채터링이 있는 가상 스위치, 4x4로 구성된 16개의 스캔 방식 가상 스위치 매트릭스를 구성하여 터치를 하여 스위치를 누를 수 있도록 하였다.

가상 디바이스는 실제 메모리에 사상된 디바이스와 같이 특정 주소에 값을 쓰거나 읽는 방법으로 동작하도록 구성되어 있다. 실험 과정에서는 이 가상 디바이스의 동작 회로를 설명함으로써, 실제 장치와 똑같은 방식으로 제어하도록 교육할 수 있다.

표 1은 가상 하드웨어 장치에 접근하기 위한 API 인데, 이 API는 통상적인 x86 CPU에서처럼

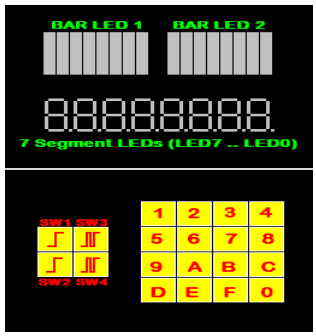

그림 6. 가상 디바이스
Fig. 6. Virtual Devices

표 1. 가상 디바이스 접근 API
Table 1. API for Virtual Devices

```
int init_virtual_io(u32 devices);
void close_virtual_io(void);
u8 readb_virtual_io(u32 addr);
void writeb_virtual_io(u32 addr, u8
value);
```

입출력 주소에 사상된 입출력 장치에 접근할 때 사용하는 입출력 함수와 똑같은 형태로 구성되며, 이 함수를 이용하여 입출력 포트에 값을 쓰거나 읽음으로써 다양한 가상 하드웨어 장치를 제어할 수 있다.

4. 이클립스 IDE

이클립스는 다양한 운영체제 플랫폼에서 사용할 수 있는 통합 개발 환경 도구이다.

본 연구에서는 프로젝트 관리, 소스 프로그램 편집, 컴파일, 다운로드 및 원격 디버깅, 버전 관리 등 개발 호스트 시스템에서 이루어지는 모든 작업을 이클립스를 통해 실행할 수 있는 환경을 제공한다.

실제 실험 환경의 구축에는 devkitPro에서 제공하는 GNU C 컴파일러 도구를 이클립스-CDT에서 사용할 수 있도록 하였다. 그림 7에서 볼 수 있는 것처럼 교육을 위한 각 실험 항목은 하나의 이클립스 프로젝트로 만들어져 각각의 Makefile을 가지도록 구성되었다. 디렉토리는 실험 관련 문서 (doc), FreeRTOS와 가상 디바이스 장치에 대한 소스 (FreeRTOS), 가상 디바이스 이미지(image), 닌텐

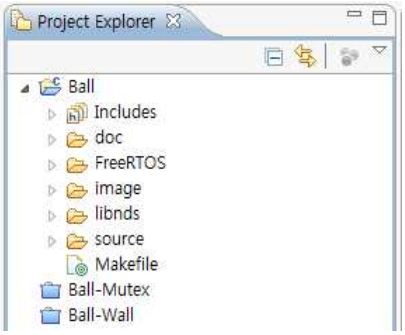

그림 7. 이클립스 프로젝트 구성
Fig. 7. Eclipse Projects

도 DS-lite의 실제 장치 접근을 위한 소스 (libnds), 그리고 실제 실험 항목에 대한 템플릿 예제 소스 (source) 등으로 구성된다.

이클립스 IDE에서는 source 디렉터리에서 작성된 각 실험 항목에 대한 소스를 FreeRTOS와 libnds등 라이브러리와 링크하여 최종적으로는 닌텐도 DS-lite의 실행 ROM 포맷인 .NDS 파일을 만들어 낸다. 이 .NDS 파일은 USB나 무선랜을 통해 다운로드 되어 타겟 시스템인 닌텐도 DS-lite에서 실행되거나, 호스트 시스템에서 실행되는 시뮬레이터를 이용하여 실행할 수 있다.

5. 다운로더

본 연구에서는 개발 호스트 시스템에서 컴파일한 결과(.NDS 파일)를 에뮬레이터가 아닌 실제 개발 장비인 닌텐도 DS-lite에 전송하여 실행하기 위한 다운로드 프로그램을 개발하였다. 다운로드 프로그램은 타겟 시스템인 닌텐도 DS-lite에서 실행되는 프로그램(그림 3의 쉘)과 개발 호스트 시스템에서 실행되는 프로그램(그림 3의 콘솔 다운로더)로 구성되어 있다. 다운로드는 닌텐도 DS-lite가 기본적으로 가지고 있는 무선 랜 상에서 TCP/IP 프로토콜을 이용한다. 다운로드는 그림 4의 하드웨어 어댑터가 있는 경우 USB 또는 시리얼 포트를 통해서도 이루어 질 수 있다. 호스트 측의 다운로드 프로그램은 리눅스 운영체제나, 윈도우즈 운영체제에서 모두 명령어 라인 인터페이스 프로그램으로 구현되어 이클립스 외부 명령 실행 설정 후, 툴바를 클릭하여 바로 구동할 수 있다. 타겟 시스템인 닌텐도 DS-lite 쪽에서는 통신 채널 선택과 해당 통신 채널에 대한 환경을 설정하고, 다운로드된 파일들에

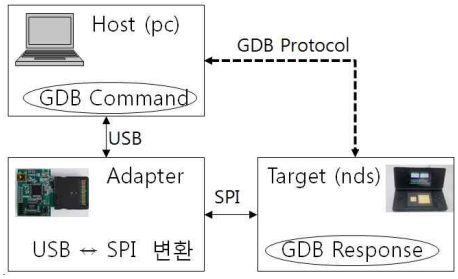

그림 8. GDB 프로토콜 전달 과정
Fig. 8. Relaying GDB Protocol

대한 관리 기능을 포함하는 쉘 프로그램이 실행된다.

6. 원격 디버깅

원격 디버깅이란 개발 호스트 시스템에서 동작하는 디버거를 이용하여 타겟 시스템에서 동작하는 소프트웨어를 디버깅하는 것이다.

본 연구에서는 원격 디버깅을 위해 GDB를 이용한다. 이를 위해 FreeRTOS가 실행되는 타겟 시스템에 GDB 스터브 코드를 이식하였다. 즉, GDB 프로토콜과 그 프로토콜에 의해 전달되는 GDB 명령들(메모리 및 레지스터 참조, 브레이크 포인트 설정, 싱글 스테핑 기능 등)을 닌텐도 DS-lite의 ARM9 CPU에 FreeRTOS 상에서 구현하였다.

개발 호스트와 타겟 시스템은 USB로 연결된다. 호스트 시스템의 이클립스에서 내려진 GDB 명령은 GDB 프로토콜 데이터로 바뀌고, GDB 프로토콜 데이터는 그림 4의 하드웨어 어댑터에 USB 인터페이스를 통해 전달되며, 다시 SPI 인터페이스를 통해 닌텐도 DS-lite에 전달된다. 이 변환 과정은 그림 8에 보였다.

이 과정을 통하여 이클립스 IDE 상에서 닌텐도 DS-lite의 FreeRTOS와 그 위의 응용 프로그램을 소스 수준에서 디버깅 가능하다.

7. 교육용 콘텐츠의 구성

본 연구에서 개발된 상용 게임기를 이용한 임베디드 소프트웨어 교육 환경은 실제 수업에 활용되었다. 수업은 장치에 대한 소개, Hello World 프로그램에 의한 개발 환경 자체에 대한 실습, FreeRTOS에 익숙해지기 위한 실험 등을 수행한 뒤, 실험 장치의 실제 하드웨어 또는 가상 하드웨어

표 2. 실험 항목
Table 2. Lab Lists

실험 제목	주요 내용
Simple IO-1	bit 입출력 (스위치, LED 접근)
Simple IO-2	State 유지가 필요한 입출력
Simple IO-3	복잡한 State machine 활용
KeyMatrix	Key Scan, 7 Seg LED 구동
KeyQueue	RTOS의 Queue 활용
Ball	격자로 움직이는 Ball 그리기
Ball-Mutex	Ball이 부딪히지 않도록 제어
Ball-Wall	가변 크기의 벽 사이를 이동

와 실시간 운영체제의 다양한 API를 활용해보는 방식으로 구성되었다.

표 2는 실제 수업에서 행해진 실험 항목과 주요 내용이다. 모든 실험은 이전 실험 결과를 활용하는 방식으로 구성되며, 실시간 운영체제의 다중 쓰레드, 쓰레드 간 통신, 상호 배제 API들에 대한 활용을 포함하여 실시간 시스템의 여러 기능들을 경험할 수 있도록 하였다.

수업 진행 결과 학생들은 주 단위로 수행해야하는 모든 실험 항목들을 문제없이 완료하여, 각종 하드웨어 제어, 실시간 운영체제의 활용 기법을 습득할 수 있었다.

수업의 마지막 단계에서는 텀 프로젝트로 간단한 게임을 만들도록 하였는데, 1, 2 인으로 구성된 팀 별로 2~3주 정도에 거쳐 간단한 닌텐도 DS-lite를 위한 게임을 실시간 운영체제의 다양한 API와 게임기가 제공하는 하드웨어를 활용하여 제작할 수 있었다. 이렇게 만들어진 게임들 역시 [12]에 공개되어 있다.

Ⅳ. 결론 및 향후 연구

본 연구에서는 임베디드 소프트웨어 교육을 위한 실습 환경이 갖추어야 할 여러 가지 요소, 즉 프로그램 다운로드 및 실행, 원격 디버깅이 가능하게 하는 게임기용 소프트웨어와 개발 호스트에서 실행되는 프로젝트 관리, 소스 프로그램 편집, 컴파일, 다운로드 및 원격 디버깅을 통합 처리하는 통합 개발 환경을 구현하였다. 연구에서는 다양한 공개 소스 소프트웨어를 적극적으로 활용하였으며, 부족한 요소들은 역시 공개 소스 소프트웨어 형태로 개발하였다. 또 교육에 필요한 추가적인 기능을 제공하기 위하여 가상 디바이스, 하드웨어 어댑터도 구현하였다.

연구에서 개발된 임베디드 소프트웨어 교육 환경은 실제 임베디드 시스템 실험 수업에 활용되어 그 효용성이 검증되었다.

본 연구는 마이크로프로세서 또는 임베디드 소프트웨어 교육 실습을 수행하기 위한 개발 환경을 구성함으로써 대학이나 전문 임베디드 소프트웨어 학원 등의 교육 기관이나 동호회, 개인들이 현실감 높은 임베디드 소프트웨어를 개발할 수 있는 환경을 매우 저렴하게 제공한다. 또 학생들이 쉽게 접근할 수 있는 하드웨어를 이용함으로써 교육에 대한 동기 부여가 가능하다.

앞으로 더욱 체계적인 임베디드 소프트웨어 교육을 위한 환경을 제공하기 위해서 추가적인 교육 실습 콘텐츠의 개발이 필요하며 본 연구에서 제공하는 환경을 통한 교육 효과에 대한 비교 연구도 필요하다.

본 연구의 모든 결과는 [8]에 공개되어 있다.

참고문헌

[1] HBE-EMPOS III-SC100 임베디드 개발보드, http://www.hanback.co.kr/products/view/99
[2] P. Varhol, "Integrated software tools improve productivity and code quality," Electronic Design, Vol. 47, No. 21, pp.62-70, 1999.
[3] S.V. Tyle, "Engineering software tools meet demands," Electronic Design, pp.71-80, 1994.
[4] 이클립스 프로젝트 홈페이지, http://eclipse.org
[5] 정창희, 우덕균, 김용상, 전인걸, 임채덕, "Nano Esto: USN 응용 소프트웨어 개발을 위한 통합 개발환경," 대한임베디드공학회논문지, Vol. 1. No. 1, pp.14-19, 2006.
[6] GBATEK 홈페이지, http://nocash.emubase.de/gbatek.htm
[7] FreeRTOS 홈페이지. http://www.freertos.org
[8] GDB 홈페이지. http://www.sourceware.org/gdb
[9] Bill Gatliff, "Embedding with GNU: the gdb Remote Serial Protocol," Embedded System Programming, November, 1999
[10] devkitPro 홈페이지, http://www.devkitpro.org
[11] Palib 홈페이지, www.palib.info/
[12] Homebrew Nintendo DS SW Development,

http://nintendo.sourceforge.net/
[13] ARM9 제품군 홈페이지,
http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.arm9/index.html

저 자 소 개

이 민 석 (Min-Suk Lee)

1986년 서울대학교 컴퓨터공학과 학사
1988년 서울대학교 컴퓨터공학과 석사
1995년 서울대학교 컴퓨터공학과 박사
1999년-2002년 (주)팜팜테크 CTO
현재, 한성대학교 컴퓨터공학과 교수
관심분야: 임베디드 시스템 및 소프트웨어, 스마트폰 플랫폼, 멀티미디어 파일 시스템, 공개 소스 소프트웨어
Email : minsuk@hansung.ac.kr