ARMulator 기반의 가상 프로토타이핑 환경

김곤O, 조상영, 이정배
한국외국어대학교 컴퓨터공학과, 선문대학교 컴퓨터공학과
{juntaO, sycho}@hufs.ac.kr, jblee@sunmoon.ac.kr

Virtual Prototyping Environment on ARMulator

Gon KimO, Sang-Young Cho, Jungbae Lee
Computer Engineering Department,
Hankuk University of Foreign Studies, Sunmoon University

요 약

프로토타입핑은 제품 개발을 위한 필요한 과정이지만 실제로 제품의 모형을 만든 후에 외형 및 기능을 검사하기 때문에 제품 개발 시간과 비용이 많이 들게 된다. 컴퓨터 기술을 이용한 가상 프로토타이핑 시스템은 이러한 단점을 보안하기 때문에 많은 연구가 되고 있다. 본 논문에서는 내장형 시스템 개발용 가상 프로토타입 플렛폼 제작을 위해 PDA와 휴대형 단말장치에서 가장 많이 사용되는 ARM코어를 기반으로 하는 ARMulator 상에 하드웨어 IP를 구현하고 실시간 운영체제인 uC/OS-II를 이식하여 내장형 소프트웨어 개발용 가상 프로토타이핑의 환경을 구축하였다. 세 개의 타스크로 구성된 검사 프로그램를 운영하여 구축된 시스템의 동작을 확인하였다. 구축된 시스템은 내장형 시스템의 소프트웨어 개발을 위한 가상 환경을 제공한다.

1. 서 론

프로토타입(Prototype)이란 제품을 생산하기 전에 만들어진 실제 크기나 축소형의 실물모델을 말한다. 제품으로 출고하기 전에 외형 및 기능을 시험하는 것은 시장 경쟁력을 갖추기 위한 중요한 생산과정 중 하나이지만 실제로 제품의[1] 외형을 만들기 위해서는 소프트웨어 또는 하드웨어를 만들어야 하므로 제품의 개발 시간이 길어지며 비용도 커지게 된다. 컴퓨터 기술의 발전으로 이러한 프로토타입을 만드는 프로토타이핑의 단점을 보완 할 수 있는 것으로 가상 프로토타이핑(Virtual Prototyping)이 가능해졌다. 가상 프로토타입은 컴퓨터 상에서 가상적으로 만들어진 프로토타입으로써 현실과 같은 제품의 외형을 제공하고 제품 외형에 붙어있는 각종 장치 및 기능이 실제와 같게 시뮬레이션 된다. 그럼으로써 실물 프로토타입과 같은 효과를 내면서도 시제품의 개발 시간단축과 비용이 절감되게 된다. 특히 제품의 리 사이클 기간이 짧은 제품에서 그 효과가 더 크다 [1,2].

본 논문은 내장형 시스템을 위한 ARM 프로세서 코어를 사용하는 제품을 위한 가상 프로토타이핑 환경 구축에 관하여 다룬다. 기존의 가상 프로토타이핑 시스템은 제품의 외형 및 기능성의 시뮬레이션의 주안점을 두고 있으나 본 논문에서의 시스템은 기존의 역할과 함께 가상의 소프트웨어 개발을 가능하도록 구현하였다.

이를 위하여 ARM사의 ARM 코어 명령어 집합 시뮬레이터인 ARMulator를 사용하여 가상 프로토타이핑 환경을 구축하였다. ARMulator에서 제공하는 기본 메모리와 타이머 환경에서 LCD 제어기와 LCD 패널 모델을 구현하여 추가하고 μ C/OS-II를 포팅 하였다. 따라서, 실제적인 고가의 개발 보드 없이도 PDA와 같은 휴대형 단말 장치 및 uC-OS-II 기반의 시스템을 개발하기 위한 가상 프로토타이핑 개발 환경을 구축하였다.

본 논문의 구성은 다음과 같다. 2절에서는 설계 및 구현에 대해 기술하며, 3절에서는 구현된 시스템의 동작에 대해 기술한다. 4절에서 결론을 맺는다.

2. 설계 및 구현

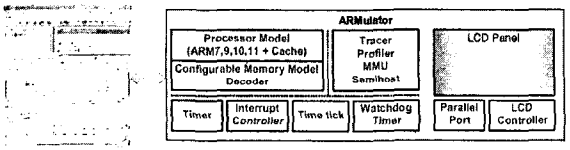
2.1. ARMulator의 확장

ARMulator는 ARM 사의 소프트웨어 개발 환경인 ADS (ARM Developer Suite) 1.2 의 한 컴포넌트로 ARM 프로세서 코어의 사이클-기반 명령어 집합 시뮬레이터를 기반으로 한 가상 보드 모델이다. 단순한 메모리 모델과 기본적인 하드웨어 IP를 가지고 있기 때문에 실제 대상 보드가 없을 경우에도 ARM 시스템을 위한 소프트웨어 개발 환경을 제공하며 소프트웨어의 프러파일링 기능을 제공하여 제품 개발 전에 하드웨어 및 소프트웨어의 성능을 예측할 수 있게 한다 [3,4].

[1] 본 연구는 대학 ITRC 육성지원사업의 연구결과로 수행되었음

본 논문에서 휴대형 단말기 제품을 위한 가상 프로토타이핑 시스템을 구축하기 위하여 ARM 프로세서 코어를 선택한 이유는 전세계 32-bit 내장형 RISC 마이크로프로세서 시장의 75%를 ARM 코어가 차지하고 있으며 대부분의 휴대형 단말기에 채택되어 있기 때문이다. 또한 계속적으로 빠르게 발전하는 ARM 프로세서를 위하여 자체 명령어 집합 시뮬레이터를 구축할 경우에 시장의 변화에 대응이 늦을 수 있기에 ARM 상의 ARMulator를 기반으로 가상 프로토타이핑 시스템을 구축하였다.

휴대형 단말기를 위한 가상 프로토타이핑 시스템을 위하여 본 논문에서는 ARMulator의 구성 및 동작을 파악하기 위하여 Parallel Port[5]를 추가하고 이를 바탕으로 그래픽 출력 응용에 사용할 수 있도록 ARM사의 A[6]에 나와 있는 소스를 참조하여 LCD 제어기와 LCD 패널 모듈을 추가하였다. ARMulator는 임의의 하드웨어 모델을 C로 기술하여 DLL 파일의 형태로 추가 확장하도록 한다. [그림 1]은 확장된 ARMulator 환경을 도시하고 있다.



[그림 1] 확장 ARMulator의 구성

ARM 사의 디버거를 사용하여 작성된 LCD 제어를 필요로 하는 프로그램을 확장 환경에 다운로드하여 타겟 보드 개발 전에 소프트웨어의 기능 및 성능을 확인하거나 하드웨어/소프트웨어 동시 설계에 활용할 수 있다.

2.2. MicroC/OS-II 이식

uC/OS-II는 RTOS (Real Time Operating System)로 Micro Controller Operating System Version 2를 말한다. 임베디드 시스템을 위한 작고 간단한 RTOS로 다양한 프로세서 아키텍쳐에 포팅 하기 쉬운 구조로 되어 있다[7]. 간단한 형태의 내장형 시스템에서 많이 사용되고 있기 때문에 uC/OS-II 기반의 가상 소프트웨어 개발 환경을 구축하기 위해서 확장된 ARMulator 환경에 uC/OS-II를 이식하였다. 이를 주요 하드웨어 IP로 타이머와 인터럽트 제어기가 필요한데 ARMulator는 2개의 16비트 다운 카운팅 타이머와 최대 32개의 자원을 제어할 수 있는 인터럽트 제어기가 있다[8].

uC/OS-II는 크게 하드웨어에 독립적인 부분, 어플리케이션에 종속적인 부분, 그리고 하드웨어에 종속적인 부분의 3가지 부분으로 나누어져 있다. 이식 시 가장 고려해야 할 부분은 하드웨어에 종속적인 부분으로서 이식에 사용된 파일 및 함수에 대해 설명한다.

OS_CPU.H

OS_ENTER_CRITICAL()과 OS_EXIT_CRITICAL()은 각각 코드의 CRITICAL SECTION을 보호하기 위해 진입 시에 프로세서 자체의 인터럽트를 금지하거나 실행의 끝에서 인터럽트 금지를 해지하는 매크로이다.

OS_CPU_C.C

1) 이 파일에서는 TCB (Task Control Block) 상의 스택 모양을 결정해준다.

2) SWI_Handler() 함수가 구현된다.

OS_cpu_a.S 의 UCOS_SWI()에 의해 호출되어 SWI (Software Interrupt) 명령어의 SWI 숫자를 확인하고 어떤 SWI 인지 판별하여 해당 처리 루틴을 호출 한다. 인터럽트 불가능과 가능을 설정하는 함수인 splx()와 임의의 인터럽트가 발생했을 때 그 해당 ISR를 등록하기 위하여 벡터를 설정하는 함수인 VectSet() 함수를 호출한다. SWI 를 통해 인터럽트를 키고 끄는 것은 사용자 타스크를 시스템 모드가 아닌 유저 모드상에서 실행 시키기 위해서 이고 ARM 프로세서는 유저 모드상에서는 인터럽트 금지를 설정할 수 없다.

OS_CPU_A.S

이 어셈블리어 파일은 하드웨어에 직접적으로 종속적인 함수들을 포함하고 있기 때문에 해당 하드웨어의 타이머 및 인터럽트 제어기에 대한 이해를 바탕으로 이식되어야 한다.

1) UCOS_IRQ (IRQ Handler)

IRQ Handler 로서 IRQ 인터럽트가 발생되면 일련의 과정을 통해서 IRQ 모드로 들어가며 0x18 번지로 하드웨어적 분기를 한다. 이 번지에 UCOS_IRQ로 다시 분기하는 명령어에 의하여 호출된다.

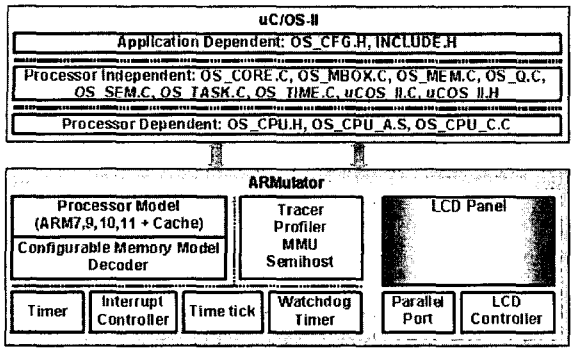
2) UCOS_SWI (SWI Handler)

SWI 핸들러는 SWI 가 발생하면 콘텍스트를 저장한 후 SWI 넘버에 따라 0x80 일 경우 콘텍스트 스위칭 함수 OSCtxSW()를 호출한다. 이 외의 경우에는 설정하여 사용하는 SWI 번호에 따라서 SWI_Handler 함수를 호출하여 정해진 번호에 해당하는 일을 수행한 후 콘텍스트를 복구하며 원래의 콘텍스트로 복귀한다.

3) OSTickISR()

이 함수는 Time Tick 타이머에 의해 발생된 인터럽트에 대해 IRQ_Handler 에서 호출되며 운영체제의 1 Tick 마다 불려져 TCB에 저장된 각 타스크들의 딜레이 값을 감소시켜준다.

[그림 2]는 확장된 ARMulator 환경에 이식된 uC/OS-II 환경을 보여준다. 이 환경에서는 uC/OS-II 기반의 응용 프로그램을 작성하여 가상의 하드웨어 및 운영체제 환경에서 동작을 시켜볼 수 있다.

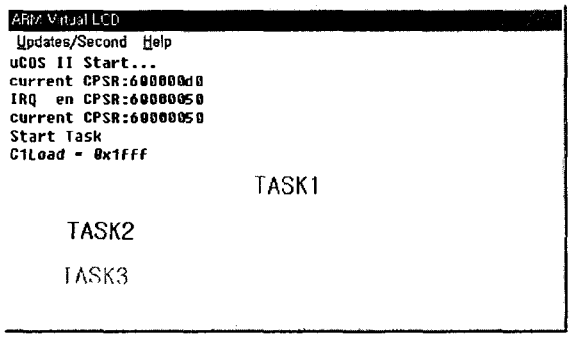


[그림 2] 전체 시스템 구성도

3. 응용 프로그램 구현 및 동작 검사

구현된 시스템의 동작 검사를 위하여 간단한 응용 프로그램을 작성하여 정상적인 이식을 확인하였다. OS의 가장 중요한 요소인 스케줄링이 정상적으로 이루어 지는 가를 테스트 하기 위하여 3개의 타스크를 priority를 차례대로 부과 하여 생성하였다. 메인에 의해 기본 적인 타스크가 생성이 되고 그 타스크에서 TASK2와 TASK3을 생성하였다.

실험은 우선 각 타스크들이 무한 루프를 돌면서 좌 우측을 왕복하며 한 픽셀씩 이동하며, 각기 다른 지연값을 가지고 있게 하였다. TASK1은 100 Tick의 지연을 가지며, TASK2는 200 Tick의 지연을 가지면서 세마포어 pend를 하게 하였고 TASK3에서 400 Tick의 지연을 가지면서 세마포어 post를 하게 하였다. [그림 3]은 실행 시의 LCD. 패널을 캡쳐한 모습이다.



[그림 3] 프로그램 실행 화면

CPSR은 프로세서의 상태를 나타내 주는 레지스터로 처음에 유저 모드와 인터럽트 금지 상태이다. SWI를 통해 인터럽트 가능 상태로 바꾸어주었다. 그 다음 Timer1의 Load값에 0x1fff 값를 넣어 타이머를 동작시킨다. 그 후 TASK1이 실행되고 TASK1이 OSTimeDly()를 통해 100 Tick동안 waiting상태로 들어가게 된다. 그 다음 TASK2가 실행되고 마찬 가지로 OSTimeDly()에 의해 200 Tick 동안의 waiting에 들어가며 그 후 TASK3가 실행된다. TASK1은 100 Tick주기로 TASK3는 400 Tick주기로 움직이기 때문에 TASK1이 4번 출력하는 동안 TASK3는 1번 출력하게 된다. 이 과정은 타이머 인터럽트에 의해서 일어나는 OSIntCtxsw이 정상적으로 작동 하는 것을 보여준다. TASK2는 200 Tick의 주기로 2번 출력 되어야 하지만 세마포어에 의해 TASK3과 동기화 되어 있기 때문에 400 Tick의 주기의 TASK3이 post를 해주지 않으면 깨어나지 못한다. 따라서 이벤트에 의한 OSCtxsw이 정상적으로 동작함을 보여준다.

4. 결론 및 향후 연구과제

본 논문에서는 ARM사의 소프트웨어 개발 환경인 ADS 1.2에 포함되어 있는 ARM 코어 명령어 집합 시뮬레이터를 이용하여 하드웨어 IP를 추가 확장하고 그 위에 RTOS인 MicroC/OS-II를 포팅하여 내장형 소프트웨어 개발용 가상 프로토타이핑 환경을 구축하였다. 구축된 환경의 동작을 검사하기 위하여 3개의 타스크로 구성된 응용 프로그램을 실행하였다. 구축된 환경에서는 LCD를 이용하는 내장형 시스템의 소프트웨어 개발 및 설계에 활용할 수 있으며 uC/OS-II 기반의 응용 프로그램을 가상의 환경에서 실행할 수 있다.

향후 다양한 내장형 시스템 개발 환경을 지원하기 위하여 하드웨어 IP 모듈을 추가하고 전체 시스템뿐만 아니라 개별 하드웨어 IP 모듈을 재구성할 수 있도록 해야 한다. 또한 재구성된 시스템의 외형을 실제 형태로 보여 줄 수 있도록 사용자 인터페이스를 구축하여야 한다.

참고 문헌

[1]http://myhome.hitel.net/~tbno/work1/VirutalProtyping.ppt

[2] 원종혁, 한상용, " 사용자 인터페이스 향상을 위한 3차원 VP시뮬레이터 설계" , HCI2000 학술대회 발표논문집, pp417-422, 2000.

[3] Steve Furber," ARM System on chip Architecture" . Addison Wesley, 2000.

[4] ARM Cop. " ARM DDI0100E ARM Architecture Reference Manual" .

[5] ARM Cop., " ARM DAI0032E Application Note32 The ARMulator" .

[6] ARM Cop., " ARM DAI0092B Application Note92 LCD and Keyboard ARMulator model" .

[7] JEAN J. LABROSSE, " MicroC/OS-II Real Time Kernel 2/E" , R&D Technical Books, 2002.

[8] ARM Cop., "ARM DDI0062D Reference Peripherals Specification".