Embedded RTOS

QEMU를 통해 임베디드 RTOS 구현하기

요약

임베디드 컴퓨터란 무엇인가?

RTOS란 무엇인가?

부팅하기

UART, 인터럽트

**작성자** 53기 문연수

[yyyynoom@gmail.com](mailto:yyyynoom@gmail.com)

1. Narrative

전문 용어가 등장하는 경우, 한글로 옮겼을 때 그 의미가 명확하지 않다면 원어 그대로 사용한다. 그 외의 모든 단어는 한글로 옮겨 작성한다. 가령 후술할 ‘hard real-time’의 경우 ‘단단한 실시간’, ‘강력한 실시간’ 등의 단어로 번역해도 그 의미가 직관적이지 않기에 논문에 나온 ‘hard real-time’ 이라는 용어를 그대로 사용했다.

일부 용어들의 경우 원어를 발음에 따라 한글로 옮겨 적은 경우가 다수 있는데 이는 해당 용어가 원어 혹은 번역된 용어보단 한글로 쓰이는 것이 관용적인 경우이다. 예를 들어 task는 그 뜻이 “a piece of work, especially something unpleasant” 이기에 ‘과업’ 혹은 ‘과제’ 라는 단어로 옮길 수 있으나 컴퓨터, 특히 운영체제에서 task는 “컴퓨터 시스템 내 활동의 기본적인 단위.” 라는 뜻으로 쓰인다. 또한 이러한 의미로 쓰이는 모든 것들을 관용적으로 태스크라 부른다. 이러한 경우 task를 원어 그대로 쓰거나 ‘과업’, ‘과제’ 등으로 옮기지 않고 한글로 ‘태스크’라 작성한다.

약어의 경우 처음 등장할 때에는 생략하지 않고 그대로 작성한다. 그 다음 해당 용어 뒤에 소괄호와 함께 ‘이하 ~(물결표)’ 로 작성한다. 이후 나오는 ‘~’에 해당하는 약어들은 해당 소괄호 앞에 붙은 용어를 의미한다. **2. Background Theory**에 나오는 ‘Operating System’의 경우 “Operating System(이하 OS)” 로 작성하였다. 해당 서술 다음 나오는 모든 약어 ‘OS’는 ‘Operating System’을 뜻한다.

1. Introduction

임베디드 시스템에서 동작하는 Real-Time Operating System(이하 RTOS)를 개발하는 것이 목적이다. 모든 동작이 언제나 정해진 시간에 수행되진 않으나, 어느 정도의 적시성을 보장하며, 사용자가 발생시킨 이벤트에 따라 적절한 동작을 수행하는 Soft Real-Time OS를 구현한다. 개발 환경은 다음과 같다:

* 개발 PC의 운영체제: Linux
* 컴파일러: ARM-GCC
* 하드웨어: RealView Platform Baseboard for Cortex-A8
* 에뮬레이터: QEMU

시계이(가) 표시된 사진

자동 생성된 설명

Figure 1 RTOS 개발 순서

개발 순서는 다음과 같다:

1. 개발 PC(Linux)에서 소스코드를 작성한다.
2. 코드를 컴파일러(GCC-ARM)를 통해 ARM아키텍쳐로 크로스 컴파일한다.
3. 생성된 바이너리 이미지를 QEMU 에뮬레이터를 통해 실행시켜 테스트를 진행한다.

1~3 순서를 반복하며 RTOS의 살을 붙여 나간다.

1. Background Theory

기본적으로 운영체제, 컴파일러, C, 어셈블리, 그리고 ARM 아키텍처에 대한 어느 정도의 지식이 있다는 것을 가정하고 해당 학술지를 작성하였다. 따라서 Context Switching, ELF, IRQ, Linker, volatile 키워드, Makefile등과 같은 용어들의 기능과 뜻에 대해서는 설명은 따로 하지 않음을 알린다.

* **임베디드 컴퓨터 (Embedded Computer)**  
   임베디드 컴퓨터는 다른 장치에 포함되어 있어서 미리 정해진 한 가지 일이나 몇 가지 소프트웨어만 실행하는 컴퓨터이다 (David A. Patterson, John L. Hennessy 2018, 5). 즉 Personal Computer (이하 PC)처럼 여러가지 기능 (음악과 동영상 재생, 게임, 문서 작성, etc.)을 수행하기보단 만들 때부터 그 기능이 한정되어 특정 기능만을 계속 수행하는 장비라는 뜻이다.   
   임베디드 컴퓨팅 장치도 소프트웨어가 있어야 동작한다. 규모가 작거나 극단적인 최적화가 필요한 임베디드 장치는 운영체제 없이 펌웨어로만 동작하기도 하지만 시스템의 자원과 복잡도를 관리하는 것이 더 중요한 임베디드 장치에서는 임베디드 운영체제를 사용하는 것이 일반적이다 (이만우 2019, 4).
* **RTOS (Real Time Operating Systems)** Operating system(이하 OS)이라는 용어에 “real-time” 이 쓰였을 때 의미하는 바는 간단히 말해 임베디드 애플리케이션이 외부 이벤트에 실시간으로 응답할 수 있음을 뜻한다. 그 반대는 다른 태스크를 기다리거나 혹은 수행이 완료될 때까지 기다리는 것을 의미한다. 이는 또 다시 ‘Hard real-Time’과 ‘Soft Real-Time’이란 용어로 나뉘게 된다.   
   hard real-time 은 그 행동이 **언제나** 정해진 마감 시간에 반드시 끝나야 함을 의미한다. 이는 비디오 스트리밍 처리, 자동차 엔진의 스파크 플러그의 점화, 도플러 레이더의 에코 처리 등을 포함한다.   
   Soft real-time 은 hard real-time 과는 다르지만 일종의 적시성을 필요로 하는 곳에 적용된다. 그 말인 즉, 마감 시간을 놓치는 것이 시스템의 무결성을 손상시키진 않으나, 해로운 효과를 가질 수 있음을 말한다. 이들은 소매점의 point of sale (POS) 시스템, ATM과 다른 신용 카드 장치, 그리고 PDA 등이 될 수 있다. (D. Stepner, N. Rajan, D. Hui 1999, 2)
* **QEMU**QEMU는 상표로 등록되지 않은, 그리고 오픈 소스인 machine emulator이자 virtualizer이다.  
   machine emulator를 사용하면, QEMU는 OS를, 그리고 하나의 장치(예, ARM 보드)에서 만들어진 프로그램을 다른 장치(예, 독자의 개인 PC)에서 구동이 가능하다. 또한 dynamic translation을 사용함으로써, 좋은 성능을 끌어낼 수 있다.  
   virtualizer를 이용하면, QEMU는 guest code를 호스트 CPU에서 직접 실행함으로써 네이티브와 가까운 성능을 달성할 수 있다. (Main Page, Aug 28, 2020)

1. Calculation or Simulation or ~ Method

시뮬레이션, 실험 등 해당 연구를 진행하는 방법에 대해 서술합니다.

4. Results

(1) 결과1 분석

~ 잘 나왔다.

(2) 결과2 분석

~ 잘 나오지 않았다. 결과를 보여주고 결과에 대해 분석한 내용을 서술합니다.

5. Conclusion

결론입니다. 위 내용들을 종합하여 한 문단에 요약하고 앞으로의 과제나 미래, 실험의 최종 결과 등으 로 내용을 끝맺습니다.

Reference

1. D. Stepner, N. Rajan and D. Hui, "Embedded application design using a real-time OS," Proceedings 1999 Design Automation Conference (Cat. No. 99CH36361), New Orleans, LA, USA, 1999, pp. 151-156, doi: 10.1109/DAC.1999.781301.
2. David A. Patterson and John L. Hennessy. *컴퓨터 구조 및 설계:하드웨어/소프트웨어 인터페이스*. n.p.: 한티미디어, 2018.
3. "Main Page." *QEMU Wiki*. last modified Jul 9, 2020, accessed Aug 28, 2020, https://wiki.qemu.org/Main\_Page.
4. 이만우. *임베디드 SO 개발 프로젝트:ARM 기반 펌웨어/RTOS의 원리와 구조*. n.p.: 인사이트, 2019.