Unicorn Engine을 활용한 Arduino 가상화를 위한 연구

이철한 안준섭 ○ 조진성

경희대학교 컴퓨터공학과
herrtane@khu.ac.kr anjs0918@khu.ac.kr chojs@khu.ac.kr

Study of Arduino Virtualization by using Unicorn Engine

ChulHan Lee JunSeop An ○ JinSung Cho

Department of Computer Science and Engineering, KhungHee University

요 약

최근 4차 산업혁명의 흐름에 맞추어 교육계에서는 초,중,고,대학교에 걸쳐서 Arduino를 통한 Embedded 소프트웨어 교육이 이루어지고 있다. 그러나 Arduino는 하드웨어가 반드시 필요하기 때문에, 하드웨어의 유무에 따른 제약이 적지 않다. 이에 본 논문에서는 Unicorn Engine을 이용하여 ARM Architecture 기반의 Arduino Due를 가상화하여 하드웨어의 제약을 받지 않고 Arduino를 구동하는 방법을 설계한다.

1. 서 론

4차 산업혁명에 따라 산업이 변화하고 있다. 이러한 상황에 맞춰 교육계에서는 정보통신 기술에 유능한 인재를 육성하기 위해 소프트웨어 교육을 진행하고 있으며, Arduino를 사용한 Embedded 소프트웨어 교육도 많이 이뤄지고 있다. 값싼 Arduino 보드를 이용하여 간단한 컴퓨터 기판에 여러 장비를 연결하여 단순한 기능을 구현할 수 있기 때문이다. 더불어 싸고 간단한 자동화 작업에 유용하기에 사물 인터넷, 로봇 및 드론, 공장 자동화 등 다양한 분야에서 쓰이고 있다.

Arduino를 사용하기 위해서는 반드시 하드웨어가 필요하다. Arduino 보드가 제어하는 다양한 작업을 프로그래밍하는 것을 목적으로 하기에, 테스트나 동작 결과를 실물로써 확인해야 한다. 그러나 최근 코로나19로 인해 비대면 일상으로 전환되면서 실질적인 Arduino 실습이 어렵게 되었다. 특히, 온라인 교육으로 전환되면서 소프트웨어 교육에 차질을 빚을 수밖에 없었고, 반도체 품귀현상으로 인해 전자제품의 가격이 올라가거나 구하기 어렵게 되었다.

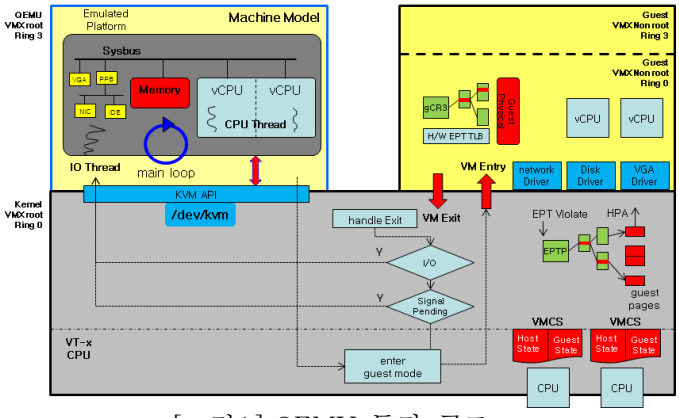
따라서 본 논문에서는 Arduino 관련 하드웨어 구매 비용과 사용환경 제약을 없애고, 작업을 테스트할 수 있는 Arduino 가상화를 목표로 한다. 최종 결과물은 Unicorn 에뮬레이션 라이브러리를 통해 아두이노 보드를 가상화할 것이다. 본 논문의 2절에서 에뮬레이션에 사용되는 Unicorn emulator와 그 기반이 되는 QEMU에 대해 설명하고, Arduino가 무엇인지 설명한다. 3절에서는 Arduino Due 보드를 대상으로 하여 가상화 과정을 제안하고, 4절에서 결론과 향후 연구 방향을 제시한다.

2. 관련 연구

2.1. QEMU

QEMU는 Quick EMUlator의 약자로, 오픈소스로 제작된 Machine 에뮬레이터이자 가상화 도구이다. 주로 시스템 에뮬레이션에 쓰이며 CPU, 메모리 등 여러 하드웨어를 가상화할 수 있다. QEMU는 호스트 OS 위에서 동작하는 hosted hypervisor이고, [그림1]과 같은 방식으로 동작한다.

QEMU는 게스트 OS로 하여 호스트의 CPU를 직접 제어하도록 프레임워크를 제공한다. 또한, User mode에서 에뮬레이션할 수도 있다. 이는 Application 수준에서 지원하는 방법으로, 원하는 시스템의 동작 에뮬레이션을 자체 명령을 통해 제공한다.



[그림1] QEMU 동작 구조

QEMU는 다양한 아키텍처와 다양한 OS를 지원한다. ARM, MIPS, PPC, RISC-V, x86-64 등 다양한 CPU 아키텍처에서 에뮬레이션을 지원한다. 더불어 Linux, MacOS, Windows, UNIX 계열 OS 등 다양한 OS에서 사용할 수 있다.

2.2. Unicorn Emulator

Unicorn Emulator는 QEMU에 기반한 가벼운 멀티 플랫폼용 CPU 에뮬레이터 라이브러리이다. QEMU와 Unicorn Emulator의 차이점은 크게 다음과 같이 구별할 수 있다.

QEMU에 비해 Unicorn Emulator는 가볍고 빠르게 동작한다. 물리적 하드웨어 전체에 대한 에뮬레이션을 제공하는 QEMU와는 달리, Unicorn Emulator는 CPU 에뮬레이션만을 목표로 하므로 불필요한 하드웨어 에뮬레이션 코드를 제거할 수 있었다.

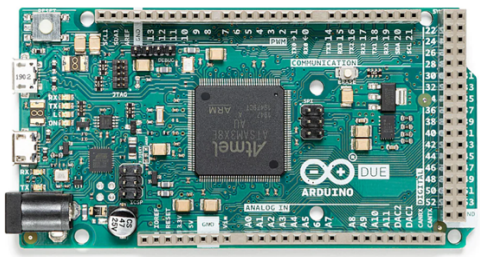
구조가 단순하며 유지보수가 쉽다. QEMU와 Unicorn Emulator 모두 다양한 아키텍처를 지원한다는 특성을 가진다. 여러 아키텍처에서 에뮬레이션을 진행해야 하므로 QEMU 내부에는 공통으로 사용되는 코드 부분이 존재한다. 즉, 여러 아키텍처를 지원하는 에뮬레이터가 공동으로 사용하는 전역변수와 내부 공유 데이터가 존재한다. 이러한 QEMU의 구조적 특징은 불필요한 빌드 시간을 늘리고, 복잡도를 증가시킨다. Unicorn Emulator는 아키텍처별로 데이터와 변수를 분리하여 의존성을 줄였다.

마지막으로 보안이 우수하다. QEMU에서 지적되었던 문제점 중 하나는 메모리 누수였다. 유니콘 에뮬레이터는 메모리 누수 문제를 해결했고, thread-safe하게 설계하여 취약점을 제거하였다.

이외에도 C, python, java, rust 등 다양한 프로그래밍 언어에서 에뮬레이션 작업을 진행할 수 있으며, 그에 따른 간단하고 직관적인 API를 제공한다. 또, JIT(Just In Time) 컴파일 방식을 사용하여 높은 성능을 제공하며, GPLv2 라이선스 방식을 통해 무료로 배포·사용할 수 있다.

2.3. Arduino

Arduino는 사용하기 쉬운 하드웨어 및 소프트웨어를 기반으로 하는 오픈소스 전자 플랫폼이다. 2005년 이탈리아에서 학생들을 위한 교육용 도구로 등장해, 현재는 초보자와 전문가가 여러 Embeded 환경에서 하드웨어를 제어할 수 있는 상호작용 기반 환경의 토대를 마련하고 있다.



[그림2] Arduino Due

[그림2]에서 볼 수 있듯, Arduino 보드는 마이크로컨트롤러와 I/O Pin, USB 포트, 전원인가 포트 등 간단한 제어 작업을 위한 컴퓨팅 시스템의 필수 구조로 되어있다. 마이크로컨트롤러의 경우, 대부분 AVR 컨트롤러가 보드 내 탑재되어 있으나, 고성능 작업을 위한 ARM 컨트롤러가 탑재된 버전도 존재한다. 프로그램의 업로드를 편리성을 위해 마이크로컨트롤러 안에 boot loader가 내장되어 있다. Arduino 프로그래밍을 위한 통합 개발환경도 제공된다. 여기에는 기본적인 편집 기능을 포함해 컴파일러, 업로딩 기능과 각 Arduino 보드에 맞는 환경 설정값을 제공한다. 또, 기타 개발에 필요한 옵션들과 라이브러리를 관리할 수 있다.

[그림3] Arduino IDE 2

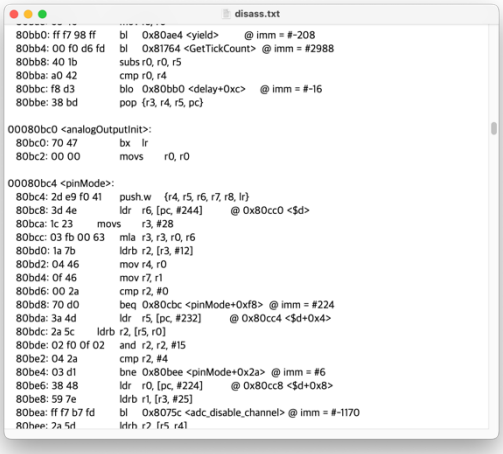
간단한 프로그래밍을 통해 마이크로컨트롤러를 통해 쉽게 동작시킬 수 있다는 것이 Arduino의 큰 장점이다. 또한 모듈의 가격 저렴하고, 자체 IDE를 제공함으로써 다양한 환경에서 쉽게 프로그래밍할 수 있다. 8bit 혹은 16bit 기반 AVR 마이크로컨트롤러를 사용하기 때문에 복잡한 작업보다 간단한 작업을 수행하는 Embeded 분야에 많이 쓰인다. 2023년 기준으로 Arduino 관련 하드웨어는 보드, 실드, 캐리어, 키트 및 기타 액세서리를 포함하여 약 100여 종류가 존재한다.

3. 시나리오 제안 및 구현

본 장에서는 Arduino 소스코드를 컴파일하여 생성된 바이너리파일을 Unicorn Engine에서 직접 Emulating함으로서, 원하는 Arduino 소스코드를 가상환경에서 구동하는 시나리오를 제안한다.

3.1. Arduino Due 기반 컴파일

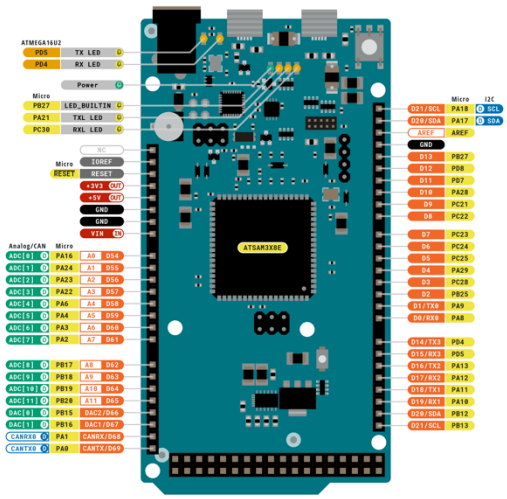
일반적인 Arduino는 AVR Architecture로 이루어져 있기 때문에, Arduino IDE로 컴파일하여 나오는 결과 바이너리 및 Hex파일을 AVR을 지원하지 않는 Unicorn에서 실행시키기는 어렵다. 따라서, 본 연구에서는 ARM32 Architecture 기반의 Arduino Due를 대상으로 한 컴파일 결과 바이너리 및 ELF 파일을 분석하여 연구를 진행하였다. 실제 컴파일 된 ELF 파일을 objdump의 disassembly도구를 사용하여 [그림 4]과 같은 ARM 어셈블리 코드 결과값이 출력되는 것을 확인할 수 있다. 바이너리 파일 역시 ELF파일과 유사하게 분석을 진행한다.



[그림 4] Objdump로 출력한 Arduino Due 기반 ELF 파일

3.2. Unicorn에서의 Arduino Pin 연결 설정

구동하고자 하는 Arduino 소스코드가 만약 LED모듈을 필요로 하는 코드일 경우, 사전에 Pin을 먼저 연결해야 한다. [그림 5]와 같이 Arduino Due에서는 GPIO기능을 지원하며, 사용자가 특정 Pin을 직접 Arduino와 연결하여 원하는 IO기능을 구현할 수 있다. 본 연구에서는 이에 주목하여 가상 모듈과의 Pin 연결 기능을 Unicorn Emulator 환경에서 구현한다. Arduino Due에 존재하는 Register, GPIO(General Purpose Input Output) 및 Pin map을 분석하여 사용자로 하여금 원하는 Pin끼리 연결을 할 수 있도록 한다.



[그림 5] Arduino Due Pin 구조도

3.3. Unicorn상에서의 Code Emulation

3.2절의 과정을 수행한 이후, 3.1절에서 분석한 바이너리 파일의 바이너리 코드를 16진수 Hex Format으로 변환하여, [그림 6]와 같이 Unicorn Engine의 Input Code로 입력하여 Emulating을 시작한다. 단, Register 초기값 등의 설정이 안되어 있으므로, 경우에 따라 Unicorn 상에서의 Register 초기화 및 세팅이 필요할 수 있다.

```
from unicorn import *
from unicorn.arm_const import *
from binascii import *
# code to be emulated
ARM_CODE32 = b'\x00\x80\x08\x20\xFD\x05\x08\x00\x9D\x0A\x08\x00\x9D\x0A\x
# memory address where emulation starts
ADDRESS = 0x10000
print("Emulate arm32 code")
try:
    # Initialize emulator in ARM-32bit mode
    mu = Uc(UC_ARCH_ARM, UC_MODE_ARM)
    # map 2MB memory for this emulation
    mu.mem_map(ADDRESS, 4 * 1024 * 1024)
    # write machine code to be emulated to memory
    mu.mem_write(ADDRESS, ARM_CODE32)
```

[그림 6] Arduino Binary Code를 사용한 Unicorn Emulating

Emulating이 정상적으로 진행되었다면, 해당 코드에 의해 LED 모듈의 몇 번째 LED가 점멸하는지 화면에 출력함으로써 사용자가 해당 코드의 동작 여부를 파악하기 쉽게 한다.

4. 결론 및 향후 연구

본 논문에서는 Unicorn Engine을 활용한 Arduino 가상화에 대한 방법을 제시하였다. 이를 통해, Arduino를 필요로 하는 교육계를 포함한 다양한 분야에서 하드웨어의 제약 없이 Arduino개발을 진행할 수 있을 것으로 기대된다.

지금까지의 본 연구에서는 Unicorn상에서 Instruction Error등의 오류가 발생했다. 따라서 향후 연구를 진행하면서 이러한 문제점을 해결하는 것이 중요할 것으로 보인다. 또한, 본 연구에서 이론적으로 제시하였던 Arduino GPIO 기능이 실제로 정상적으로 작동되도록 프로그램을 구현할 계획이며, 결과적으로 LED 모듈의 작동 등을 가시화하여 화면에 출력하는 기능에 대한 연구를 진행할 것이다.

참고문헌

[1] Unicorn Engine Presentation: https://www.unicorn-engine.org/BHUSA2015-unicorn.pdf
[2] Unicorn Engine Documentation: https://www.unicorn-engine.org/docs/tutorial.html
[3] QEMU Documentation: https://www.qemu.org/docs/master/
[4] Arduino Official Site: https://www.arduino.cc/en/Guide/Introduction
[5] NGUYEN AnhQuynh and DANG HoangVu, “Unicorn : Next Generation CPU Emulator Framework”, 2015