안드로이드 호스트 기반 KVM/ARM 가상화 환경에서의 GPU 디바이스 구동기법

박병수 ○, 이운성, 박세진, 박찬익, 김우성*, 조혜진*, 이규은*
포항공과대학교, LG전자*
{bspark○, dnsrjd1212, baksejin, cipark}@postech.ac.kr,
{rain.kim, haejin.cho, kyueun.yi}@lge.com

Driving GPU Device of Android in KVM/ARM Virtualization Environment

Byungsu Park, Unsung Lee, Sejin Park, Chanik Park, Woosung Kim*, Haejin Cho*, Kyuenu Yi*
POSTECH, LG Electronics*

요 약

스마트폰으로 대변되는 ARM 프로세서 기반 모바일 환경에서, 보안성 지원, 복수개의 실행환경 지원 등을 위해 최근 가상화 기술이 도입되고 있다. 이에 ARM 프로세서는 Virtualization Extension(VE) 라는 하드웨어 가상화 기술을 지원하고 있으며, 소프트웨어 환경으로, KVM/ARM, Xen on ARM 등의 하이퍼바이저가 연구 중이다. 하지만 현재 ARM 환경에 적용 가능한 하이퍼바이저와 함께 설치되는 리눅스 커널에서는 GPU 디바이스 드라이버가 제한적으로 지원되며, 다양한 GPU 드라이버가 지원되는 안드로이드 플랫폼에 설치되는 하부 리눅스 커널 버전은 ARM/KVM 하이퍼바이저를 지원하지 못하는 문제를 가진다. 본 논문에서는 이러한 ARM 가상화 환경에서의 GPU 디바이스 관련 개발 환경의 한계점을 극복하고자, GPU 디바이스가 구동되는 안드로이드 모바일 플랫폼 환경에서 KVM/ARM을 하이퍼바이저로써 구동시키는 방법을 제안한다.

1. 서 론

가상화 기술은 최근 서버 환경뿐만 아니라 스마트폰으로 대변되는 ARM 프로세서 기반 모바일 환경에서도 널리 연구되고 있다. 다중의 게스트 단말을 하나의 물리적 단말에서 독립적으로 구동시켜, 이들 간의 고립을 지원해 주는 모바일 가상화 기술은 최근 대두되고 있는 모바일 보안 문제[6,7]를 해결하기 위한 핵심기술로써 각광 받고 있다. 가상화 환경은 서버 통합(Server consolidation) 등 주로 서버 환경에서 높은 가용성을 위해 적용 되어 상대적으로 GPU관련 연구가 활발하지 않았다. 그러나, 모바일 환경은 GUI, 3D 모바일 게임등 GPU 사용이 극대화 되는 환경이므로, 가상화 환경 GPU 디바이스를 직접 사용 할 수 있어야 한다. 하지만 최근까지 연구 된 모바일 가상화 환경에서는 GPU 디바이스를 지원하는 환경이 극히 제한적이다. 본 논문에서는 이러한 ARM 가상화 환경에서의 GPU 디바이스 관련 개발 환경 제약을 극복하고자 GPU 지원 안드로이드 운영체제를 KVM/ARM 하이퍼바이저 호스트 운영체제로 사용할 때 필요한 문제들을 정리하고 있으며, 실제 임베디드 장치상에 구현할때 필요한 실제적인 이슈를 정리한다.

2. 기초 지식 및 관련 연구

ARM 프로세서는 가상화 지원을 위해 ARMv7부터 Virtualization Extension (VE)을 탑재하고 있다. 세부적으로 VE 기술은 최고 특권 모드(Privilege Level 2)인 Hyp 모드와 Second stage page table 등이 존재한다. Hyp 모드는 가상화 지원을 위해 새롭게 정의된 모드로써, Non-secure 상태에서 동작하며, Second stage page table 은 게스트 운영체제의 효과적인 메모리 가상화를 위해 지원되고 있다.

하이퍼 바이저 관련 연구는 다음과 같다. KVM/ARM[4]은 ARM 프로세서에서 KVM 하이퍼바이저를 구동시켜주는 연구로써 현재 GPU 디바이스를 지원하지 못하고 있다. 현재 가용한 모바일 ARM 임베디드 장치의 경우, 리눅스 커널 3.4를 기반으로 GPU를 포함한 I/O 장치 드라이버가 배포되고 있으나, KVM/ARM은 리눅스 버전 3.9 이후부터 지원하고 있는 상황이라, KVM/ARM 에서 GPU를 구동시키기가 어려운 상황이다. KVM/ARM 뿐만 아니라, 현재 활발히 연구가 진행중인 Linaro[2], Xen on ARM[1] 등 ARM 프로세서 기반의 가상화 환경에서도 GPU장치가 구동되지 못한다. 이는 GPU 드라이버의 구동 조건과 하이퍼바이저의 구동 조건이 다른 상황에서 기인한다.

3. KVM 호스트화 과정에 필요한 요소

3.1 Hyp 모드 지원

KVM/ARM은 호스트 운영체제의 부팅 과정에서 초기화가 진행 된다. 이 때, 시스템이 제공해야 할 조건으로 ARM CPU 모드 및 상태가 Hyp 모드/Non-secure 상태여야 한다. KVM/ARM은 호스트 커널 이미지가 최초 램에 올라 갔을 때, 기본적인 하이퍼바이저 백터 테이블 설정 및 하이퍼바이저 관련 컨트롤 레지스터들을 설정하는데, 이 때 필요한 접근 권한이 Hyp 모드 및 Non-secure 상태 이다. 때문에 KVM/ARM을 지원하는 부트로더는 호스트 커널 이미지를 램에 올리기 전에 반드시 Hyp 모드/Non-secure 상태로 동작해야 한다.

3.2 Large Physical Address Extension (LPAE) 지원

ARM 프로세서는 ARMv7부터 LPAE[3]를 지원하기 시작했다. LPAE는 40bit의 물리적 어드레스를 제공하면서 최고 1TB의 물리 메모리를 사용할 수 있게 해준다. 이는 다수의 게스트를 운영해야 되는 하이퍼바이저에게 유용한 설계이며, ARM의 경우, VE와 LPAE를 반드시 함께 사용해야 한다. 즉 KVM/ARM 하이퍼바이저가 운영되기 위해서는 호스트 운영체제는 반드시 LPAE를 사용해야 된다.

3.3 커널 API 호환성 제공

1장에서 언급했듯이, KVM/ARM은 리눅스 커널 버전 3.9부터 포함되어 있다. 이와 같은 이유로 하위 리눅스 커널 버전 에서 KVM/ARM을 운영할 시, 버전 차이에 따른 커널 API 호환성을 고려해야 한다. 나아가 버전 차이가 상당히 많은 경우, API 뿐만 아니라 서브 시스템 변화에 따른 대응 역시 필요하다.

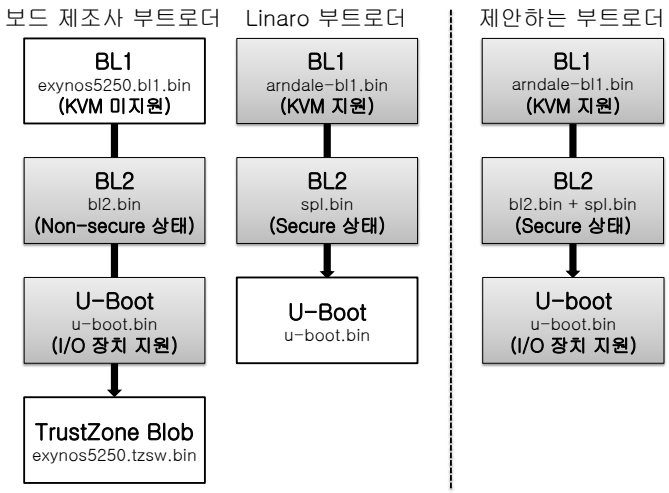

그림1. 안드로이드 KVM 호스트를 위한 부트로더의 구성. 제안 된 부트로더는 Linaro 에서 제공한 BL1(Secure 상태로 BL2 로딩) 과 Hyp 모드 변환 코드가 삽입 된 BL2, U-boot로 구성됨.

4. 안드로이드 플랫폼에서의 KVM 하이퍼바이저 구성

본 논문에서는 Arndale-5250 임베디드 보드를 사용하여 안드로이드 운영체제의 KVM 호스트화를 진행 하였다. Arndale-5250 보드는 ARMv7 Cortex-A15 1.7GHz Exynos 5250 dual core, 2GB 램 및 Mali-T604 3D가속 그래픽 하드웨어를 지원한다.

4.1 부트로더

그림1의 좌측 부분은 Arndale-5250 보드 제조사가 기본적으로 제공하는 부트로더를 나타낸다. 부트로더는 BL1, BL2, U-boot, TrustZone Blob 으로 구성 되어 있으며, 이 가운데 BL1은 바이너리 형태로 제공 되고 있기 때문에 관련 필요한 기능을 수정하거나 추가할 수 없다. 부트로더는 호스트 커널 이미지를 램에 올리기 전에, CPU 모드를 최고 특권 모드인 Hyp 모드 상태로 바꿔야 하지만 보드 제조사가 제공하는 BL1의 경우, Non-secure 상태로 BL2를 로드한다. BL2가 Non-secure 상태라는 의미는 CPU 모드를 Hyp mode로 바꾸기 위해 필요한 동작들을 상태 권한 제한으로 실행 할 수 없음을 뜻한다. 때문에 보드 제조사가 제공하는 BL1은 사용할 수 없다. 본 논문에서는 위의 문제를 해결하기 위해 ARM 지원 단체인 Linaro 에서 제공하는 Arndale-5250용 BL1을 사용하였다. 그림1의 가운데 부분과 같이 Linaro 제공 BL1은 ARM의 Secure 모니터 백터 기능을 제외하여, BL2를 Secure 상태로 로드한다.

그림1의 우측 제안하는 부트로더는 Linaro BL1 과 Secure 상태로 로드 되어, CPU 모드 변환 코드가 삽입 된 BL2를 보여준다. Secure 상태로 로드 된 BL2는 특권 상태를 갖고 있어, 아래와 같은 CPU 모드 변경 절차를 수행할 수 있다.

첫째로, Non-secure 상태를 위한 인터럽트, Non-Secure Access Control Register (NSACR) 레지스터를 셋팅하고, 둘째로는 모니터 백터 테이블 어드레스 및 핸들러 함수를 설정 한다. 셋째로 U-boot 단계에서 Secure Monitor Call(SMC) 호출을 통해 해당 핸들러에 진입하여, Hyp Vector Base Address Register(HVBAR) 설정 및 현재 상태를 Non-secure 상태로 변경 한다. 이 후 마지막으로 HVC 호출을 통해 CPU 모드를 Hyp 모드로 전환 후, 커널 이미지를 로드 한다.

4.2 커널 & KVM/ARM

4.2.1 LPAE 지원

호스트 운영체제는 LPAE를 반드시 지원해야 한다. 하지만 Arndale-5250 제공 커널 설정의 경우, LPAE가 비활성화되어 있다. 또한 이를 활성화 할 시에, 현재 호스트 운영체제 커널 버전에서는 unaligned 주소 공간에 대한 메모리 매핑 처리가 완벽하지 않기 때문에 문제가 발생한다. 관련 패치 코드 [5] 삽입으로 LPAE를 작동 시킬 수 있다.

4.2.2 TrustZone 관련 설정 제거

Arndale-5250 보드의 경우, ARM 프로세서의 Secure 모니터 백터 기능을 이용한다. 이를 위해 BL1에서 이미 모니터 백터 초기화 설정을 마무리 하고, BL2를 Non-secure 상태로 로드한다. 현재 시스템에서는 Linaro 제공 BL1을 사용하였기 때문에, 기존 모니터 백터 기능을 사용할 수 없다. 하지만 호스트 운영체제에서 모니터 콜 (SMC) 을 이용한 기능은 어떤 동작에 대해 모니터 콜을 이용해서 서비스를 받느냐 안 받느냐의 차이므로, 커널 설정에서 비활성화 시킴으로써 모니터 백터 기능을 사용할 수 없는 상황을 해결할 수 있다.

4.2.3 KVM/ARM 설치 및 호환성

본 논문에서의 KVM/ARM 코드는 리눅스 버전 3.9를 기반으로 하였다. 하지만 Arndale-5250 보드 제조사 제공 커널 버전이 3.4.5 이기 때문에, KVM/ARM에서 사용하는 커널 API와 현재 호스트 운영체제 커널이 사용하는 커널 API가 서로 호환가능한지 고려해야 한다. 다행히도, 현재 버전으로 KVM 호스트화 진행 결과, 일부 함수에서만 호환성에 문제가 있었고, 대부분의 커널 API들이 하위 버전 커널과 호환을 이루었다. 또한 메모리 서브 시스템 정책 등 역시도 큰 변화가 없었음을 확인 하였다.

5. 평가 및 향후 계획

(a) 안드로이드 (호스트) (b) 2개 게스트 화면전환

(c) 안드로이드 (게스트) (d) 우분투 (게스트)

그림 2. 안드로이드 호스트 KVM 에서 두 개의 게스트 운영체제 (안드로이드, 우분투) 구동 화면 스크린 샷. 각 게스트 운영체제는 VNC 클라이언트로 접속하여 화면 출력됨.

KVM에서 게스트 구동 평가를 위해 [8]에서 제공하는 QEMU/ARM 버전을 일부 수정 하여 사용하였다. 그림 2는 GPU 디바이스가 구동되는 안드로이드 모바일 플랫폼 환경에 KVM호스트화를 완료한 상태들로써 각각의 게스트들을 VNC 클라이언트를 통해 나타내고 있다. 각각의 게스트는 안드로이드 JB-MR1, 우분투 12.04 버전 운영체제를 사용하였고, 2개의 VNC 클라이언트 프로그램을 통해 동시에 두 개의 게스트를 운영하였다. 두 개의 게스트 운영체제 구동이 문제 없이 작동 됨을 확인 할 수 있었다. 또한, 기존 KVM/ARM 연구로 구동 시에는 임베디드 보드의 내장 LCD 등의 장치가 활성화 되지 않아, 터미널 접속만 가능한 상태였지만, 본 연구의 결과로, 호스트 운영체제인 안드로이드에서 GPU 를 비롯한 LCD 드라이버 등이 정상적으로 동작하여, 임베디드 보드의 내장 LCD 를 통해 화면이 출력되는 것을 확인할 수 있었다. 이를 바탕으로 게스트 운영체제에서도 GPU 등의 장치를 직접 사용 할 수 있는 연구를 진행 할 예정이다.

6. 결 론

본 논문은 기존 모바일 가상화 환경에서의 GPU 디바이스 구동 제약을 극복하고자, GPU 디바이스가 구동되는 안드로이드 모바일 플랫폼 환경에서 KVM/ARM 하이퍼바이저를 구동 시킬 수 있는 요소 기술들에 해서 설명하였으며, 실제 Arndale-5250 임베디드 보드에서 KVM/ARM을 직접 구현하며 발생하는 문제들을 해결 하는 방법을 제시하였다. 실제 이를 위해 부트로더부터 커널의 일부분까지의 수정이 필요하였으며, 이를 통해 복수개의 게스트 운영체제가 무리 없이 구동 됨을 확인할 수 있었다.

7. Acknowledgement

<참고문헌>

[1] Hwang, Joo-Young, et al. "Xen on ARM: System virtualization using Xen hypervisor for ARM-based secure mobile phones." Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE. IEEE, 2008.

[2] http://www.linaro.org/

[3]http://www.hotchips.org/wp-content/uploads/hc_archives/hc22/HC22.23.220-1-Brash-ARMv7A.pdf

[4] Dall, Christoffer, and Jason Nieh. "KVM/ARM: the design and implementation of the linux ARM hypervisor." Proceedings of the 19th international conference on Architectural support for programming languages and operating systems. ACM, 2014.

[5] ARM: LPAE: Fix mapping in alloc_init_section for unaligned addresses, http://lists.infradead.org/pipermail/linux-arm-kernel/2013-March/155617.html

[6] http://fninside.hyundaicapital.com/408

[7] http://kr.aving.net/news/view.php?articleId=203687

[8] https://github.com/virtualopensystems/qemu