연성 실시간 Xen 하이퍼바이저를 위한 VCPU 할당 정책 설계

허경우, 고영웅
한림대학교 컴퓨터공학과
e-mail: rapperkw, yuko@hallym.ac.kr

Design of VCPU Allocation Policy for Soft Real-time Xen Hypervisor

Kyung-Woo Hur, Young Woong Ko
Dept of Computer Engineering, Hallym University

요 약

본 논문에서는 Xen 가상 머신에서 연성 실시간 처리를 제공하기 위하여 하이퍼바이저(Hypervisor)의 VCPU 할당 정책을 제시한다. 현재 Xen 에서는 가상의 CPU (VCPU)를 스케줄링하여 전체 시스템을 관리한다. 본 연구에서는 실시간 태스크를 지원하는 VCPU 할당 정책의 사전 작업으로 기존의 Xen VCPU 할당 정책을 분석한다. 분석된 할당 정책의 단점으로 멀티프로세서 환경에서의 불필요한 VCPU 마이그레이션(Migration) 으로 인한 오버헤드를 확인하고 기존 정책의 단점을 보완하기 위한 새로운 할당 정책을 제시하여 Xen 플랫폼에서 연성 실시간을 지원하는 VCPU 할당 정책을 제안하고 있다.

1. 서론

컴퓨터 시스템의 발전과 각 모듈의 복잡성으로 인한 관리, 환경에 대한 인식 변화에 따른 그린 IT[1] 등으로 인해 최근 가상화[2] 기술이 대두 되고 있다. 가상화 기술은 하나의 하드웨어에 가상의 레이어를 두어 레이어 위에서 여러 개의 운영체제나 모듈 등이 동작하는 기술을 말한다. 하나의 하드웨어로 여러 개의 운영체제나 모듈 등을 운용할 수 있기 때문에 자원의 효용성 증대나 관리가 용이하여 다양한 분야에서 연구가 활발히 진행 중이다.

특히 임베디드 분야에서 가상화 기술 활용을 위한 연구가 활발히 진행되고 있지만 임베디드 분야의 중요한 특성인 실시간 시간 처리 기능을 가상화 기술이 제공해야 한다는 어려움이 있다[3]. 기존의 가상화 연구는 주로 하드웨어를 파티셔닝 하는 방법을 이용했기 때문에 경성 실시간 시스템에 적용되어 왔다. 하지만 임베디드 분야에서 동적으로 유입되는 태스크를 처리하는 경우가 많기 때문에 기존의 가상화 기술로는 지원하기 어렵다.

본 연구에서는 오픈 소스 가상화 프로젝트인 Xen[4]에서 연성 실시간 기능을 제공하기 위해 고려해야 할 오버헤드를 VCPU 할당 정책에서 찾아보았으며 실시간 처리를 가능하게 하기 위한 새로운 VCPU 할당 정책을 제안한다.

1) 본 연구는 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업(2010-0005442) 과 지역혁신인력양성사업으로 수행된 연구결과임

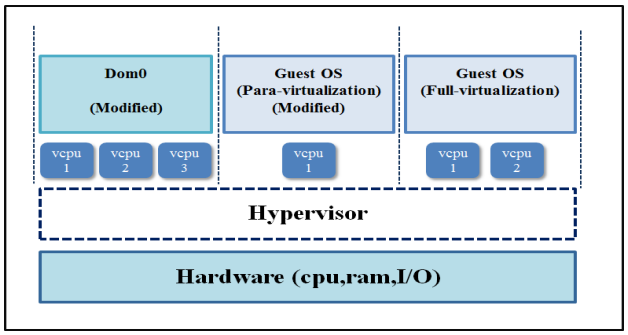
2. 관련 연구

실시간 처리를 지원하는 가상화 기술로는 현재 KUKA RTOSWin[5], RTS Hypervisor[6] 등이 대표적이다. KUKA RTOSWin은 GUI를 처리하는 부분과 실시간 적인 처리를 하는 부분을 각각 다른 운영체제가 처리한다. GUI를 처리하는 운영체제는 Windows XP/Vista를 지원하고 실시간 처리는 Windows CE, RT/Linux 등을 지원한다. 각 운영체제는 VmfWin(Virtual Machine Framework For Windows) 라는 가상 머신 위에서 수행되며 실시간 운영체제와 GUI를 담당하는 Windows 운영체제 간에는 KUKA VMF Binary Module을 통해 통신을 한다. 이것은 실시간 운영체제에서 실시간 처리가 필요한 태스크 집합에 대해서 처리를 하고 GUI를 처리해야 하는 데이터는 VMF를 통해서 Windows 운영체제로 보내주는 방법을 사용한다. KUKA RTOSWin 이 사용하는 방식은 실시간 처리를 위한 운영체제가 고정적으로 파티셔닝 되어 있는 하드웨어 자원을 사용하기 때문에 시스템 설정이 동적으로 바뀔 수 있는 상황에서 적용하기 힘든 단점이 있다. 이와 같은 방식들은 실제 산업현장에서 사용하는 고정된 실시간 태스크를 가지고 있는 시스템에는 적용하여 사용할 수 있지만 범용적으로 사용되기는 어렵다.

3. Xen 시스템 분석 및 VCPU 할당 정책

오픈소스 기반의 가상화 소프트웨어인 Xen에서의 스케줄링은 가상의 CPU (VCPU)를 이용하여 스케줄링을 한다. VCPU는 Xen 하이퍼바이저에서 게스트 운영체제에게 제공하는 가상의 CPU로서 게스트 운영체제 당 32개까지 할당할 수 있다. 하이퍼바이저 위에서 실행되는 게스트 운

영체제는 본래의 시스템에서 동작되는 것처럼 VCPU를 CPU로 인식하여 실행되며 하단에 있는 하이퍼바이저가 게스트 운영체제에 할당한 VCPU를 제어하면서 전반적인 시스템에 대한 스케줄링을 한다.



[그림 1] XEN 아키텍쳐

그림 1에서 보는 것과 같이 Xen 아키텍처의 특징은 하이퍼바이저가 마이크로커널 형식으로 메모리에 적재되고 Dom-0 는 그 위에서 각 게스트 운영체제의 요청을 받아서 처리한다. 그렇기 때문에 Xen의 시스템 부팅 순서는 하이퍼바이저가 메모리에 로딩되고 그 이후에 Dom-0 를 위한 자료구조를 생성하고 Dom-0를 다시 모듈 형태로 하이퍼바이저 위에 생성한다.

시스템이 부팅될 때 Xen 하이퍼바이저는 게스트 운영체제를 위한 VCPU를 각 CPU 코어에 할당한다. 제일 처음 생성되는 게스트 운영체제인 Dom-0는 기본으로 현재 시스템에 장착된 CPU 코어의 개수만큼 VCPU가 할당된다. VCPU는 각 CPU 코어에 맵핑되거나 스케줄링 정책에 의하여 다른 CPU 코어로 마이그레이션 될 수 있다.

Xen 에는 게스트 운영체제에게 할당되는 VCPU를 스케줄링 하기 위해 기본적으로 2개의 스케줄러가 등록되어있다. SEDF와 Credit 스케줄러[7][8]이며 Credit 스케줄러가 기본 스케줄러로 설정되어 VCPU를 스케줄링 한다. Credit 스케줄러는 Round Robin 방식의 일종으로 각 VCPU에게 우선순위와 Credit 값을 부여 하게 된다. 이 우선순위는 Credit 값의 잔여 여부에 따라 BOOST, UNDER, OVER 상태로 나뉘게 된다. UNDER 상태는 각 CPU 코어의 런큐에서 Credit 값이 남아있어 실행되기를 기다리는 상태이며 OVER는 작업을 모두 마치고 Credit 값이 충전되기를 기다리는 상태이다. 또한 BOOST는 I/O 등 인터럽트로 인해 Credit을 모두 소진 하지 못한 채 다시 스케줄링 되기를 기다릴 경우 다른 VCPU 보다 빠르게 다시 스케줄링 될 수 있는 우선순위 상태이다. Credit 값은 30ms 마다 한 번씩 활성화 되어있는 모든 VCPU 에게 충전시켜주며 하나의 VCPU가 할당받을 수 있는 Credit의 최대값은 300 이다. 각 VCPU의 Periodic 타이머가 현재 실행중인 VCPU의 할당받은 Credit 값 100을 감소시키고 30ms 마다 다시 Credit을 충전한 후 잔여 Credit 값과 우선순위로 문맥전환을 할지 결정한다.

3.1 VCPU 초기 할당 방식

Xen 하이퍼바이저에서는 게스트 운영체제를 생성할 때 먼저 도메인 자료구조와 VCPU 자료구조를 생성한다. 게스트 운영체제 당 하나의 도메인 자료구조가 할당되며 사용자의 설정에 따라서 여러 개의 VCPU를 가질 수 있는 구조로 되어있다. 도메인에 속해 있는 VCPU는 시스템의 각 코어마다 보유하고 있는 런큐에 삽입되고 Credit 스케줄러를 이용하여 각 코어의 VCPU를 스케줄링 한다.

Xen 시스템은 VCPU를 게스트 운영체제에 할당할 때 여러 가지를 고려하여 각 코어에 할당을 한다. 기본적으로 현재 시스템에서 사용가능한 CPU 코어에 할당된 VCPU의 개수를 검사한다. 확인한 각 코어의 VCPU 개수를 비교하여 제일 적게 할당되어있는 코어에 새로운 VCPU를 할당한다. 이 때 현재 시스템이 Hyper Threading을 사용하고 있다면 원래의 코어 개수보다 2배 많은 CPU 코어 개수를 가지고 있다. Hyper Threading으로 생성된 CPU 코어 ID 는 실질적인 CPU 코어 ID 보다 높은 번호로 설정이 되기 때문에 최대한 높은 ID를 가진 코어에 VCPU를 할당한다. 이는 제일 사용빈도가 적은 CPU 코어에 할당하기 위함이다.

하지만 현재 Xen 의 VCPU 할당 방식에는 큰 문제점이 있다. 바로 CPU 코어의 사용률을 고려하지 않았기 때문이다. Xen 에서는 단순히 코어에 할당되어있는 VCPU의 개수 만을 계산하기 때문에 동일한 숫자의 VCPU 가 각 코어에 할당되어 있다고 하더라도 각 VCPU가 사용하는 상이한 CPU 사용률로 인해 균등한 로드 밸런싱을 보장할 수 없다. 이러한 경우는 실시간 처리를 위한 태스크들이 수행되고 있을 때 처리를 방해할 수 있는 요인이 될 수 있다.

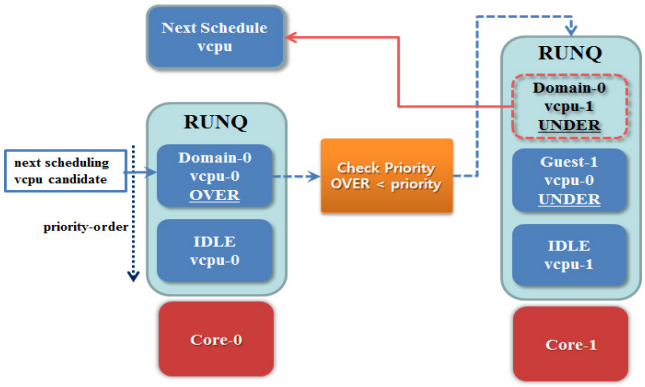
3.2 VCPU 로드 밸런싱

Credit 스케줄러의 장점은 로드 밸런싱 기능이다. 멀티프로세서 시스템에서는 각 코어 마다 할당되어있는 VCPU가 있으며 각 코어는 독립적인 런큐와 스케줄링 타이머를 가진다. Credit 스케줄러는 각 코어에 사용률을 균등하게 하기 위해 로드 밸런싱 기능을 지원한다. Xen 에 포함되어있는 SEDF 스케줄러에는 로드 밸런싱이 없다.

그림 2에서와 같이 각 코어에 VCPU가 있다고 가정한다. Core-0 에는 Xen에서 가장 먼저 생성되는 IDLE 도메인의 VCPU-0 와 Domain-0 의 VCPU-0을 런큐에 가지고 있다. Core-1 에는 IDLE 도메인의 VCPU-1, Domain-0 의 VCPU-1, GuestOS-1 의 VCPU-0를 가지고 있다. 각 런큐는 우선순위대로 정렬되어 있으며 우선순위는 UNDER, OVER, IDLE 순으로 정렬된다.

Credit 스케줄러는 Core-0 에서 다음에 수행될 VCPU를 가지고 오기 위해 런큐의 최상위에 있는 Domain-0 의 VCPU-0을 가지고 온다. 그리고 가져온 VCPU-0의 우선순위를 검사하여 UNDER 우선순위를 가지면 VCPU-0 를 수행하기 위해 스케줄러 함수로 보낸다. 그리고 스케줄러

함수는 문맥전환을 하여 Domain-0의 VCPU-0를 수행한다. 하지만 가져온 VCPU-0가 OVER 상태이면 현재 Core-0 에 수행되어야 할 VCPU가 없고 idle 상태이기 때문에 Credit이 충전되기 전까지 Core-0은 쉬게 된다.



[그림 2] VCPU 로드 밸런싱 예

이는 시스템 효율성을 감소시키기 때문에 이렇게 쉬는 것을 방지하기 위해 Credit 스케줄러는 다른 Core 인 Core-1 의 런큐를 검사한다. Core-1 런큐의 최상위부터 검색을 하여 VCPU-0 보다 우선 순위가 높은 Domain-0 의 VCPU-1를 발견하면 Core-0으로 VCPU-1를 스틸(Steal) 한다. 그리고 가져온 VCPU-1를 다음에 수행할 VCPU로 정하여 스케줄러에 넘겨준다. 로드 밸런싱 기능을 수행하면 시스템에 장착되어있는 Core들이 균등하게 VCPU를 수행하고 이는 시스템 효율성면에서 뛰어난 효과를 보인다.

3.3 VCPU 할당 정책 예비 실험(Credit 스케줄러)

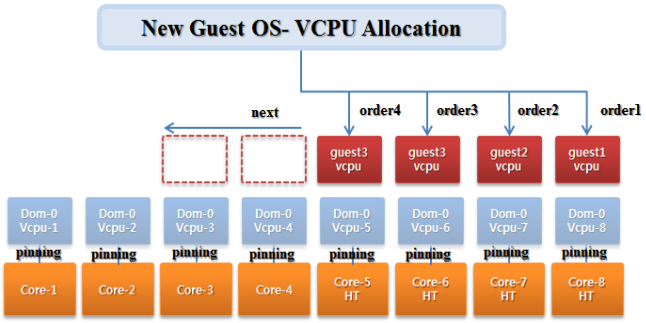
본 논문에서는 실시간 처리를 지원하고 균등한 로드 밸런싱을 제공하는 VCPU 할당 정책을 제안하기 위해서 먼저 기존의 하이퍼바이저의 VCPU 할당 정책을 확인하였다. 그 방법으로 아래의 표1의 환경에서 다음과 같은 실험을 진행하였다.

<표 1 > 실험 환경

Hardware	
CPU	Intel Xeon E5520 2.27Ghz Quad Core - HT Enable
RAM	6 GB
Software	
OS	CentOS 5.5
Xen Hypervisor	3.4.3
Guest OS	CentOS 5.4(Para-Virtualization)

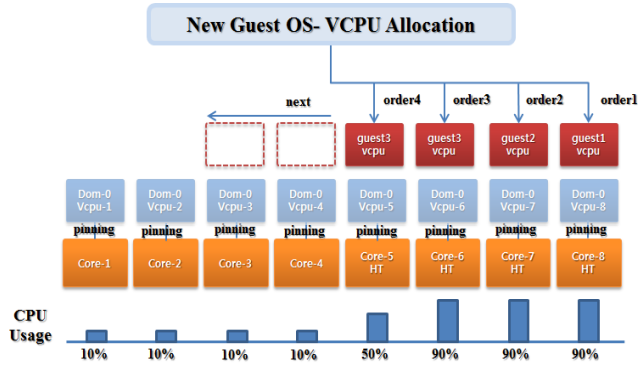
먼저 현재 시스템은 쿼드 코어 CPU 에 Hyper Threading을 활성화 시켰기 때문에 총 8개의 코어가 있는 것으로 Xen 하이퍼바이저가 인식한다. Dom-0 는 기본적으로 시스템에서 인식한 코어의 개수 만큼 VCPU를 할당받기 때문에 8개의 VCPU를 가진다. 그리고 Credit 스케줄러가 로드 밸런싱 기능을 이용하여 VCPU 를 마이그레이션 하는 것을 방지하기 위해 각각의 코어에 하나씩 pinning 한다. pinning을 하게 되면 로드 밸런싱에 관계없이 VCPU가 해당 CPU 코어에서 계속적으로 수행된다. 그리고 첫 번째 실험으로 VCPU 한 개를 사용하는 게스트 운영체제를 하나씩 계속 생성하면서 VCPU가 할당되는 CPU 코어의 ID를 확인하였다.

두 번째 실험으로 8개의 게스트 운영체제를 생성하고 VCPU를 1개씩 할당하였다. 8개 게스트 운영체제의 VCPU도 첫 번째 실험의 Dom-0 VCPU와 같이 스케줄러의 로드 밸런싱으로 인한 마이그레이션을 피하기 위해 각 코어에 하나씩 pinning 하였다. 그리고 특정한 코어들의 사용률을 일정 이상 상승시키기 위해 계속적으로 CPU 부하를 주었다. 그리고 이후에 생성되는 게스트 운영체제의 VCPU 가 어느 코어에 할당되는 지를 확인하였다.



[그림 3] 실험1 - VCPU 개수 기준 할당 순서

첫 번째 실험에서는 각각의 코어가 가지고 있는 VCPU 가 1개씩이며 모든 코어의 VCPU 개수가 동일하기 때문에 코어 ID 가 가장 높은 코어-8 에 먼저 게스트 운영체제1의 VCPU 를 할당하게 되는 것을 볼 수 있다. 그 다음에 생성되는 게스트 운영체제도 마찬가지로 VCPU 개수를 확인하고 동일했을 때 제일 높은 코어 ID를 가진 코어에 VCPU를 할당한다.



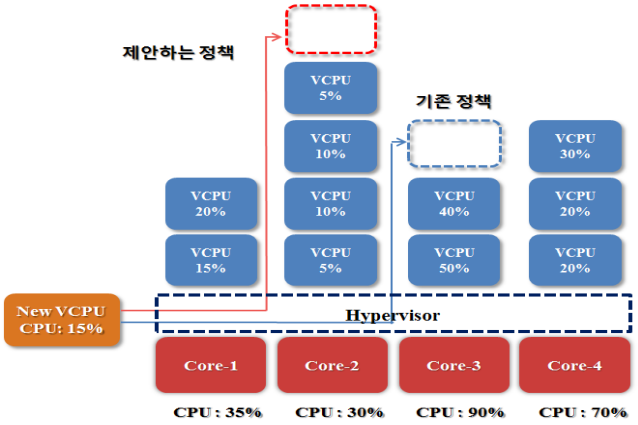
[그림 4] 실험2 - CPU 사용률 변화시 할당순서

두 번째 실험에서는 각 CPU 코어의 사용률을 변화했음에도 불구하고 CPU 코어 사용률에 관계 없이 실험1과 동일한 결과를 보여주고 있다. 이는 각 코어의 로드에 따른 밸런싱을 하는 것이 아니라 VCPU의 개수로 로드를 예측하기 때문이다. 이러한 결과는 불필요하게 새로운 VCPU를 부하가 가중된 코어에 할당할 수도 있게 되며 Credit 스케줄러의 로드 밸런싱 기능으로 또 다시 VCPU를 마이그레이션 해야 하는 상황이 발생 될 수 있다. 실시간 처리

를 위한 시스템에서는 이러한 오버헤드로 데드라인을 놓칠 수 있는 요인이 되기 때문에 실시간 시스템 내에서 최대한 배제를 해야 한다.

3.4 제안하는 VCPU 할당 정책

본 연구에서는 실험에서 확인한 단점을 보완하기 위해 새로운 Xen 하이퍼바이저 VCPU 할당 정책을 제안한다. 현재에 구현되어 있는 할당방식은 단순히 각각의 CPU 코어에 할당되어진 VCPU 의 카운트를 보고 새로운 VCPU를 할당하기 때문에 코어의 사용률을 알 수 있는 방법이 없다. 그래서 실시간 처리를 하고 있는 VCPU의 수행을 방해 할 수 있기 때문에 다음 그림과 같은 시스템을 제안한다.



[그림 5] 제안하는 VCPU 할당 정책

Xen에서 게스트 운영체제의 실시간 태스크를 보장하기 위해서는 여러 가지 요인들을 고려해야 한다. 먼저 게스트 운영체제에서 처리하는 실시간 태스크와 비실시간 태스크가 요구하는 CPU 사용률을 분석해야 한다. 분석된 사용률을 토대로 하이퍼바이저는 게스트 운영체제가 요구하는 사용률을 코어의 사용률에 맞추어 VCPU를 균등하게 할당해야 한다. 또한 게스트 운영체제는 할당받은 VCPU를 이용하여 실시간 태스크와 비실시간 태스크를 적절하게 스케줄링해서 실시간 태스크를 처리할 수 있어야 한다.

고려해야 하는 여러 가지 요인들 중에 본 논문에서는 1차적으로 게스트 운영체제가 실시간 태스크와 비실시간 태스크를 처리하기 위해 필요한 CPU 사용률을 기준으로 VCPU 할당 정책을 제안한다. 그림5와 같이 쿼드코어 시스템에 각각의 VCPU가 할당되어 있고 하이퍼바이저는 각 게스트 운영체제가 필요한 VCPU 사용률을 저장하고 있다. 새로운 VCPU가 할당될 때 기존의 Xen 하이퍼바이저는 VCPU 개수와 높은 코어 ID를 사용하는 정책을 사용하기 때문에 코어3에 할당을 하게 된다. 하지만 코어3에서 수행되야 할 VCPU의 사용률은 90%이고 현재 VCPU가 요구하는 CPU 사용률은 15%이기 때문에 하나의 코어에서 수행될 수 있는 능력을 초과한다.

제안된 시스템의 VCPU 할당 정책은 기존의 VCPU가 사용하는 사용률을 알고 있고 새로 할당될 VCPU의 요구량도 알고 있기 때문에 각 코어에서 가장 적게 CPU 사용률을 요구하는 코어에 VCPU를 할당한다. 그래서 그림에서와 같이 VCPU의 개수가 많더라도 코어2에 할당 하게 된다. 본 논문에서 제안하는 VCPU 할당 정책을 사용하면 현재 VCPU에서 처리하는 실시간 태스크와 비실시간 태스크에 영향을 주지 않고 기존의 할당 정책보다 좀 더 효율적으로 코어를 사용할 수 있으며 불필요한 마이그레이션을 줄일 수 있을 것이다.

4. 결론 및 향후 연구

가상화 기술의 연구가 활발하게 진행됨에 따라 다양한 분야에서의 활용되고 있다. 특히 임베디드 분야에서 하드웨어 성능이 높아지고 복잡한 기능을 제공하는 모듈들을 제어해야 하는 상황에서 가상 레이어를 이용하여 각 모듈을 관리하기 쉬운 가상화 기술의 활용도가 높을 것으로 예상된다. 이에 본 연구에서는 오픈소스 가상화 프로젝트인 Xen에서 실시간 스케줄링을 제공하는 프레임워크를 구현하기 위한 사전 분석으로 현재 Xen 하이퍼바이저가 VCPU를 할당하는 정책을 살펴보고 정책의 단점을 보완할 수 있는 새로운 할당 정책을 제안하였다.

본 실험 결과에서 확인할 수 있듯이 불필요한 VCPU의 마이그레이션을 줄여서 실시간 처리에 방해가 되는 요인을 최소화 할 수 있을 것으로 기대된다. 향후 연구로는 Xen에서 실시간 태스크와 비실시간 태스크를 가지고 있는 게스트 운영체제의 VCPU를 실시간 스케줄링 할 수 있는 시스템을 개발 할 것이다.

참고문헌

[1] Junghyun Shin, "그린 IT 기술 동향", 한국정보과학회, 정보과학회지, 제27권 제11호(통권 제246호) 2009.11

[2] Gil Neiger, Amy Santoni, Felix Leung, Dion Rodgers, Rich Uhlig. "Intel® Virtualization Technology: Hardware support for efficient processor virtualization". Intel® Technology Journal. 2006

[3] Junghan Kim, Inhyuk Kim, Chang-woo Min, Young Ik Eom, "모바일 가상화 기술 동향", 한국정보과학회, 정보과학회지, 제28권 제6호 2010.6

[4] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, Andrew arfield. "Xen and the art of virtualization". Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003.

[5] http://virtualization.kuka-rtos.com/en/rtos/

[6] http://www.real-time-systems.com/real-time_hypervisor

[7] http://wiki.xensource.com/xenwiki/CreditScheduler

[8] L Cherkasova, D Gupta, A Vahdat, "Comparison of the three CPU schedulers in Xen", Performance Evaluation Review, 2007, HP