Архитектура VirtualBox

Николай Иготти

Идеологические моменты

- Высокопроизводительный эмулятор x86 на x86
- Использование динамического транслятора (на основе QEMU) при необходимости
- Использование динамической модификации привилегированного кода гостя, исполняемого в R1 (зачем?)
- Использование аппаратной виртуализации при возможности
- Модели устройств частично в R0/GC
- Поддержка vSMP

Проблемы производительности

- Привилегированные инструкции и системные вызовы (GC->R0->R3->GC)
- Управление памятью (различные режимы адресации: битность, PAE, large pages)
- Обработка прерываний (внешних, внутренних)
- Обмен информации между R0/GC/R3 компонентами гипервизора
- Ввод-вывод, особенно видеопамять, диск и сеть

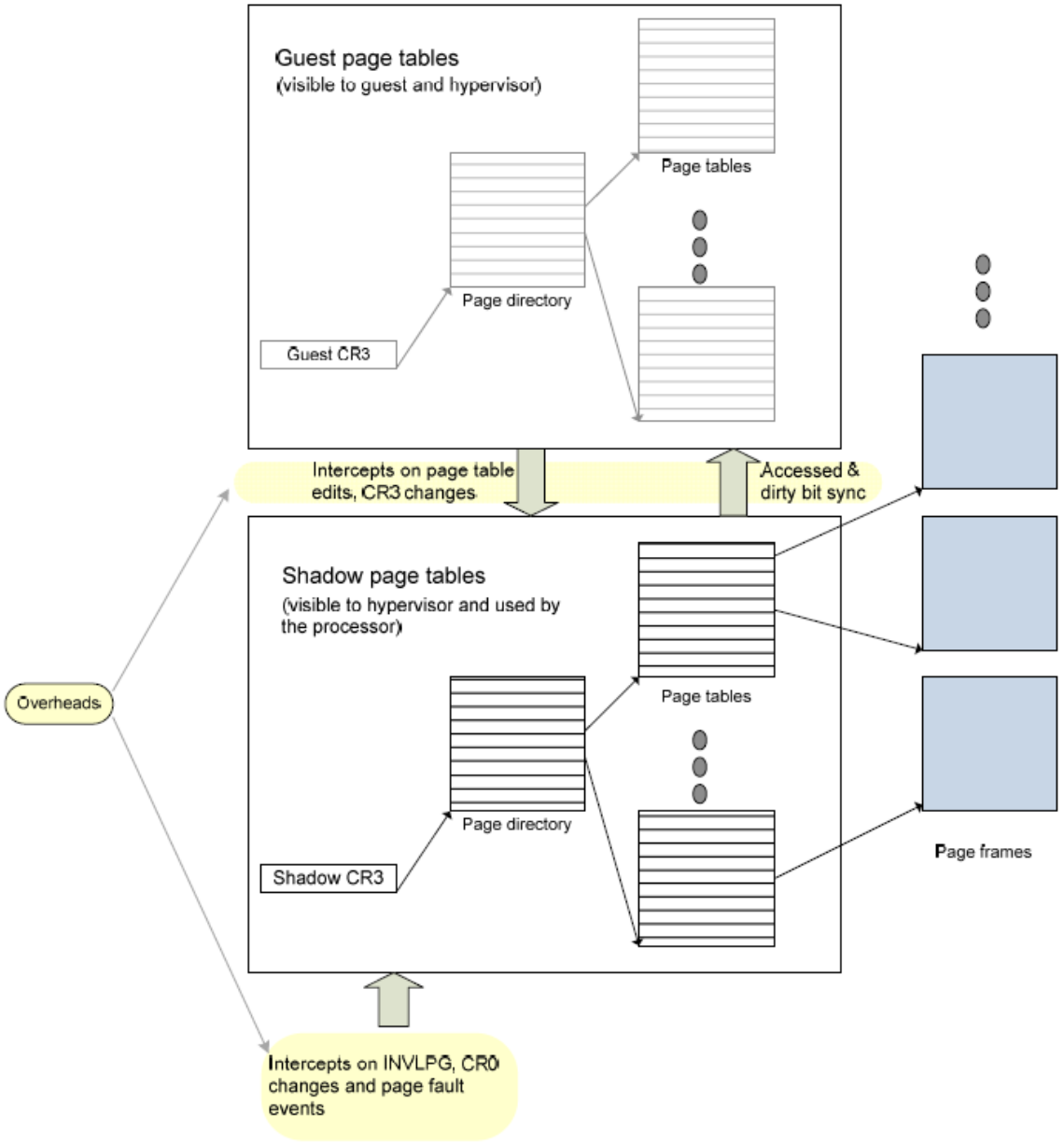
Подходы к решению

- Максимум непосредственного исполнения
- Интенсивное использование MMU процессора (защита чувствительных страниц)
- Диспетчеризация прерываний прямо в R0 (GC(03)->R0->GC(03))
- Значительная часть кода гипервизора в ядре
- Или в контексте гостевой ОС (в кольце 1)
- Код в GC полностью релоцируемый
- Динамический поиск и модификация участков кода ядра гостя с привилегированными инструкциями

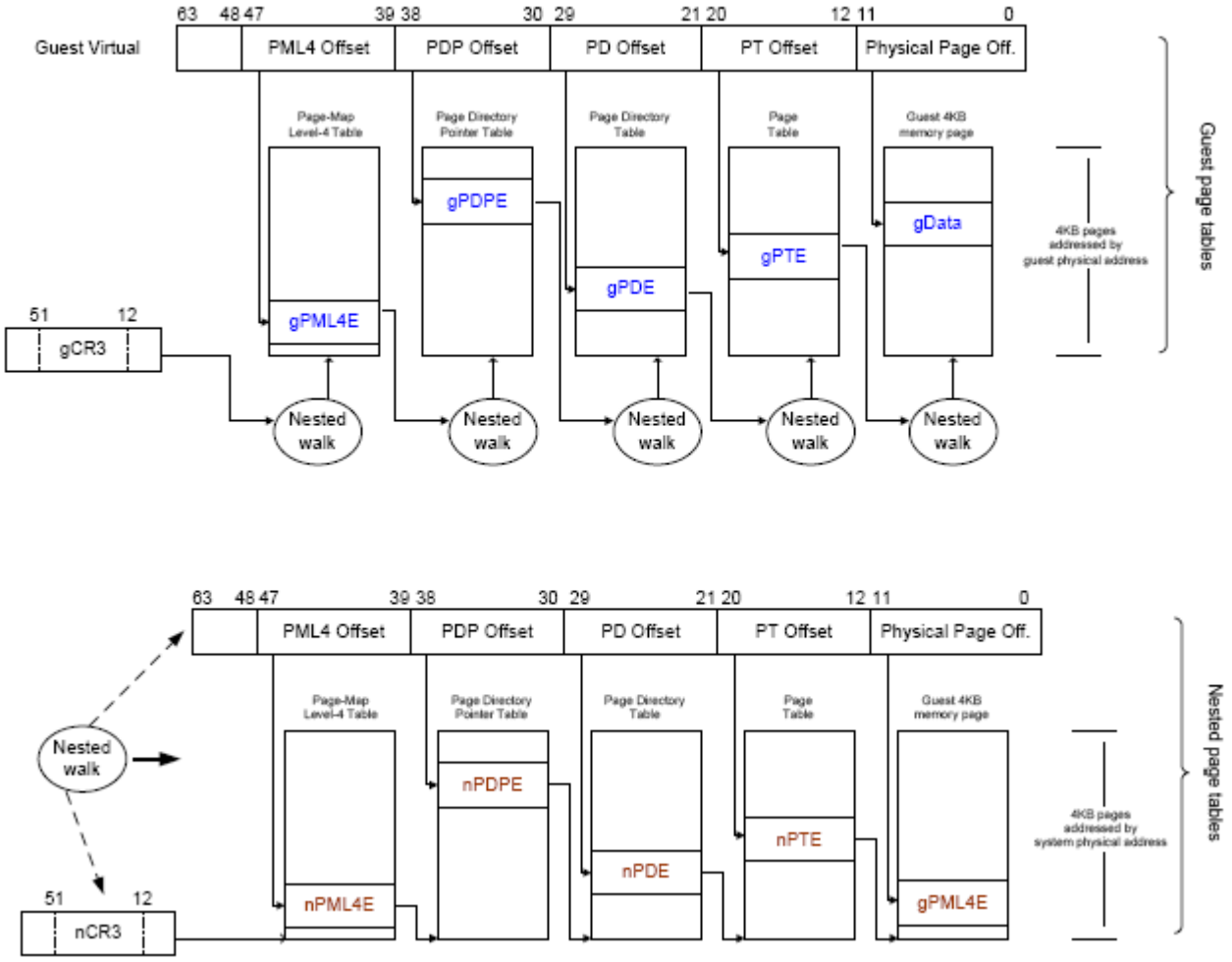
Подходы к решению (2)

- Явное разделение кода модели устройства на R0/GC/R3 части
- Сложная логика в R3, часто исполняемый код в R0/GC
- 16-битный код исполняется динамическим транслятором
- «Ленивое» выделение физической памяти
- Теневые или вложенные таблицы страниц, в зависимости от возможностей MMU процессора

Теневые таблицы страниц



Вложенные таблицы страниц*



* Взято из документации AMD

Организация ввода/вывода

- Обработка привилегированных операций и MMIO из обработчиков исключений
- Самобалансирующееся AVL дерево обработчиков с ключом – физическим адресом/размером
- “Впрыск” прерываний в гостевую ОС
- Паравиртуализованные драйвера
- Оптимизация доступа к VESA фреймбуфферу
- Доступ к реальным устройствам (USB, проброс PCI)

Вопросы, информация

- Пожелания по курсу
- Экзамен, записывайтесь на igotti@gmail.com