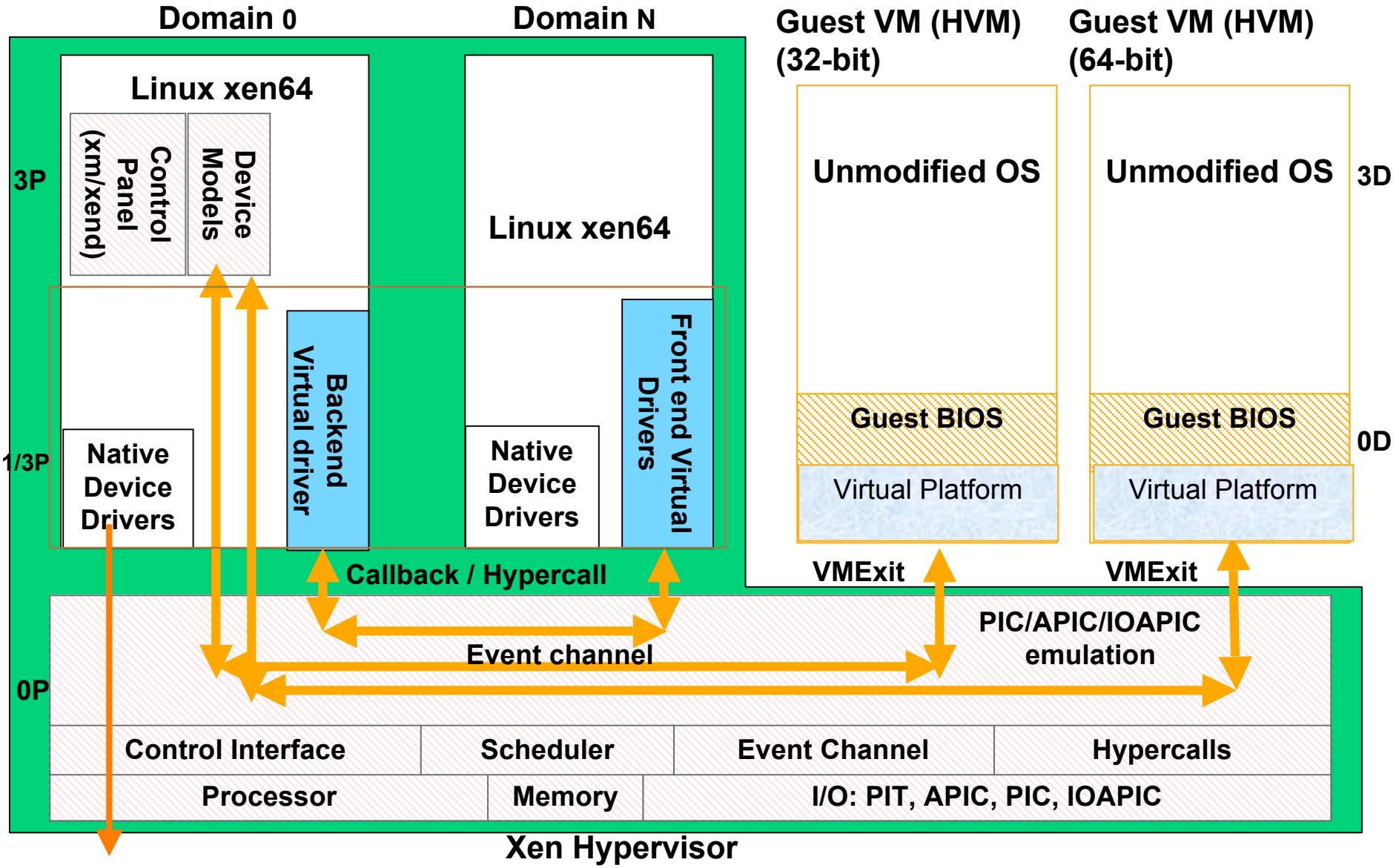
Progressive paravirtualization

Keir Fraser, XenSource

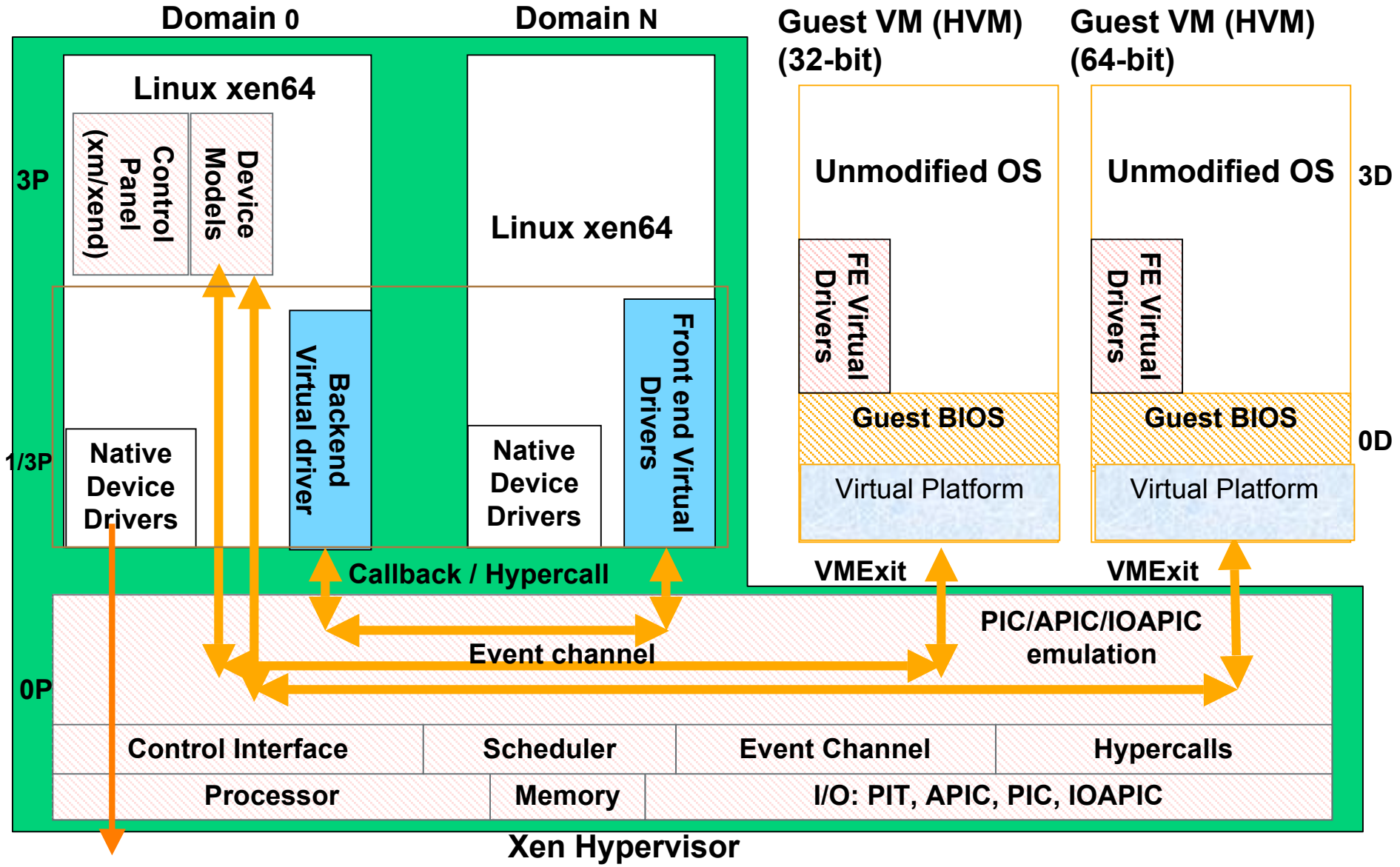
HVM Architecture



Progressive paravirtualization

- Hypercall API available to HVM guests
- Selectively add PV extensions to optimize
  - Net and Block IO
  - XenPIC (event channels)
  - MMU operations
    - multicast TLB flush
    - PTE updates (faster than page fault)
  - Time
  - CPU and memory hotplug

PV Drivers


Domain 0
Domain N
Guest VM (HVM) (32-bit)
Guest VM (HVM) (64-bit)
Linux xen64
Linux xen64
Unmodified OS
Unmodified OS
3P
3D
Control Panel (xm/xend)
Device Models
FE Virtual Drivers
FE Virtual Drivers
Guest BIOS
Guest BIOS
0D
1/3P
Native Device Drivers
Backend Virtual driver
Native Device Drivers
Front end Virtual Drivers
Virtual Platform
Virtual Platform
Callback / Hypercall
VMExit
VMExit
PIC/APIC/IOAPIC emulation
Event channel
0P
Control Interface
Scheduler
Event Channel
Hypercalls
Processor
Memory
I/O: PIT, APIC, PIC, IOAPIC
Xen Hypervisor

Hypercalls

- HVM guest can detect hypervisor platform via CPUID instruction
  - New hypervisor leaves at 0x40000000
  - Look for signature 'XenVMMXenVMM'
  - Space for future expansion and feature flags
- Hypercall page is filled in by writing its address to a special MSR
  - Location determined via CPUID
  - Currently always MSR 0x40000000
- Hypercall page hides low-level details of transferring control to the VMM

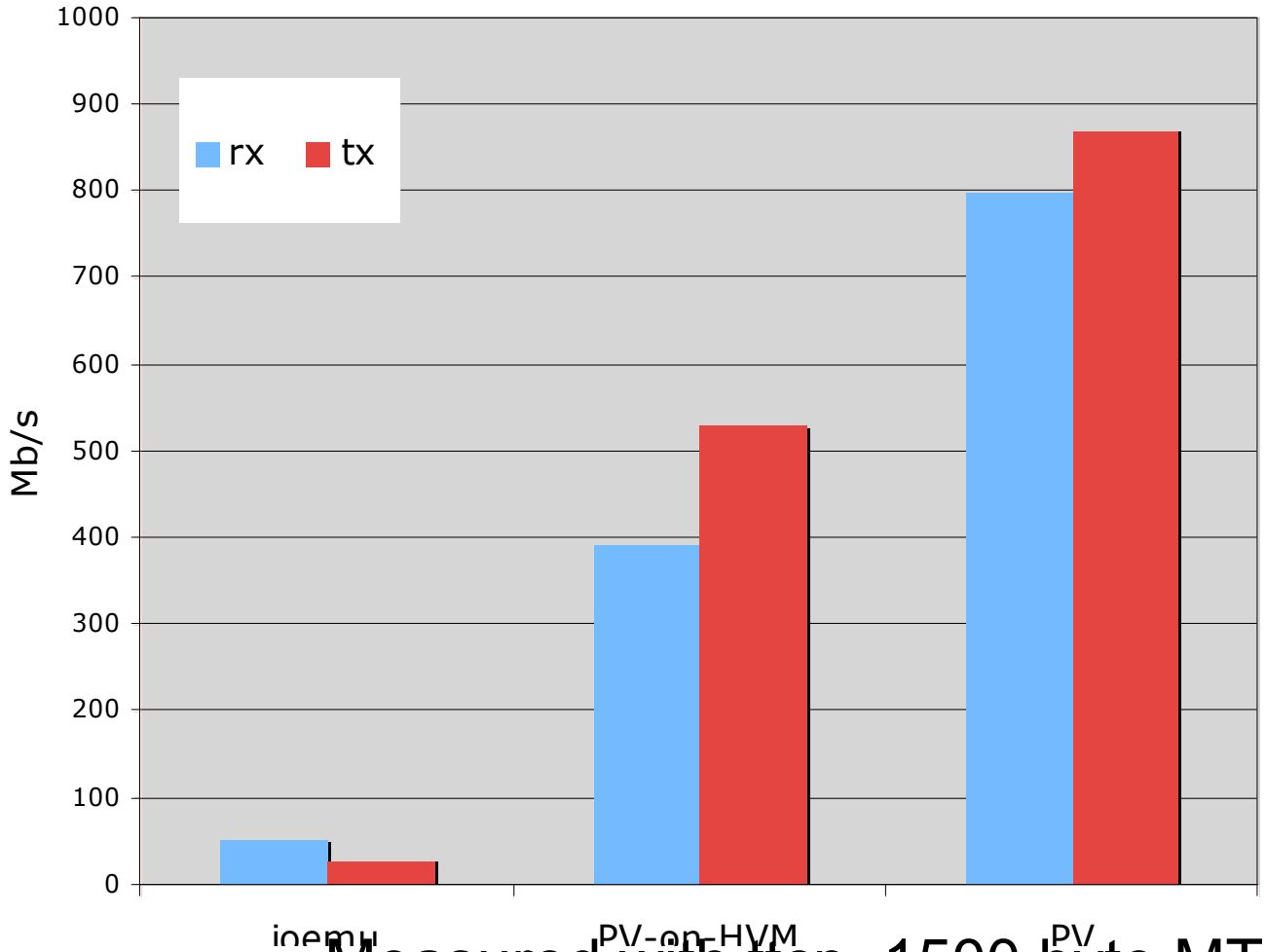
Building PV drivers for HVM

- PV drivers depend on architectural features of Xen
  - Grant tables for memory sharing
  - Event channels for asynchronous notifications
- Encapsulate support in a 'platform driver'
  - Ioemu defines a dummy PCI device that triggers loading of the platform driver in the HVM guest
- Xenbus, blkfront, netfront can be built as separate modules against a native Linux build
  - See unmodified_drivers/linux-2.6 in the xen-unstable tree

Xen support for PV-on-HVM

- Event-channel notifications cause an interrupt to be delivered via the virtual APIC on a pre-registered vector
  - 'Platform driver' registers itself on that IRQ and demuxes pending events to registered drivers
  - Future: IRQ per-device or per-VCPU
- Virtual CPUID and MSR addresses to allow hypervisor detection and hypercall setup
- Hypercalls are being incrementally extended to support HVM guests
  - Also require support for 32-bit guests on 64-bit hypervisor: the 32-bit and 64-bit ABIs are different
  - This work overlaps strongly with PAE-on-64 PV guest support

PV Driver performance



Measured with ttcp, 1500 byte MTU