AMD64 Status Report for Kernel Summit

Rich Brunner, AMD Fellow

AMD

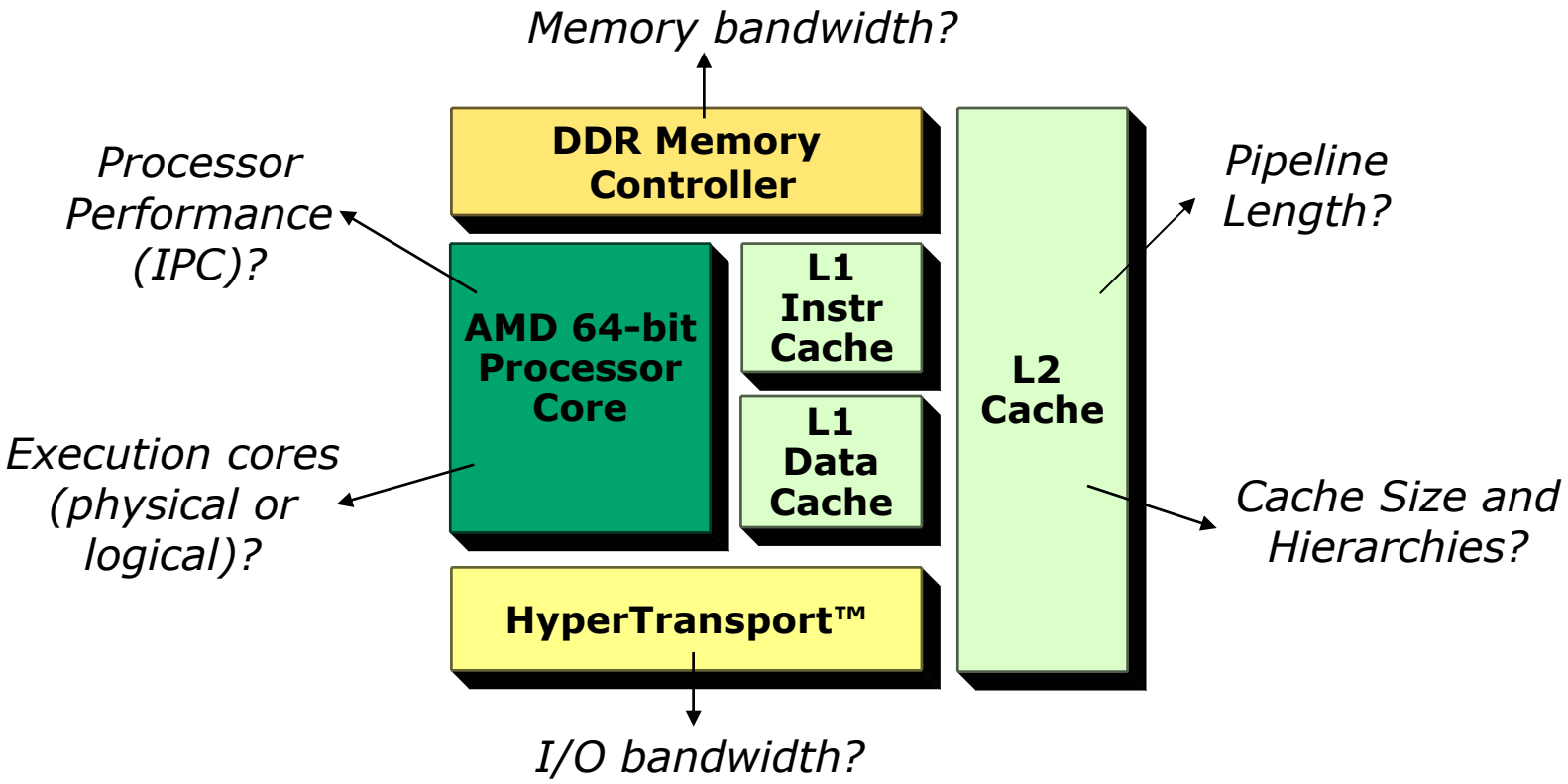
Linux has become a proving ground for 64-bit computing

- AMD64 becomes a mainstream architecture supported by the Linux kernel and leading vendors:
  - Thanks to many other kernel hackers
  - Linux distributions and OEMs embrace AMD64
  - More than 15 Linux distributions and community projects offer AMD64 versions . . . and this support is growing!
  - Not just for the server market, for desktop as well
- AMD continues to support various FOSS organizations and conferences with dollars and hardware
  - OSDL, FSG, GCC conference, OKS/OLS
- AMD continues a trend of openly providing early technical information on our products to the developer community for feedback . . . "

AMD Dual Core

A Word from Our Sponsor …

The size, energy consumption, & performance needs of today's computers require semiconductor makers to use new innovations in product design

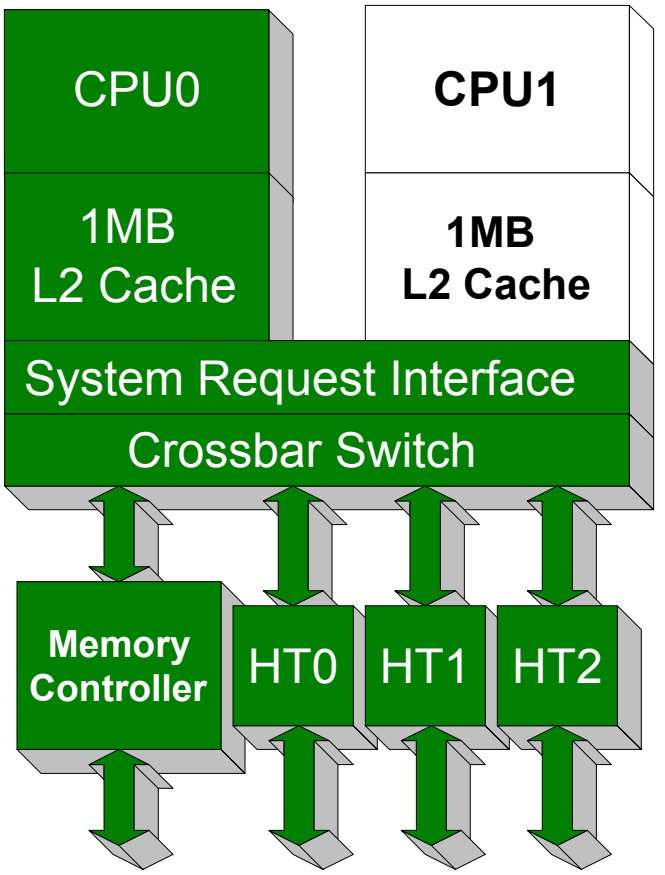


Dual core processor technology allows AMD to continue to offer a competitive performance roadmap while meeting the system architecture demands of our customers

Introducing AMD64 Dual Core Processor

Processors based on AMD64 with Direct Connect Architecture were designed from the start to add a second core

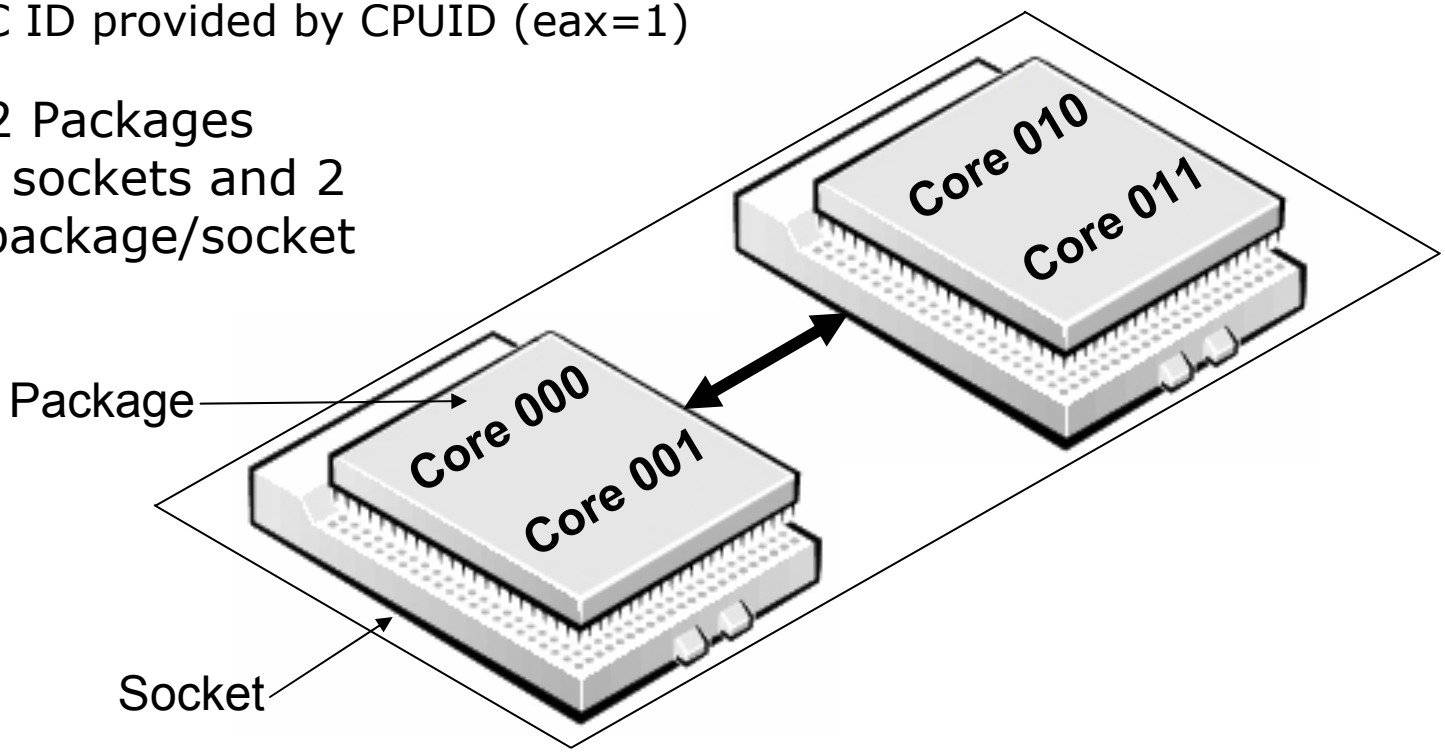
- Dual core processor will be in AMD's forthcoming 90nm process
- Expected availability in mid-2005
- One die with 2 CPU cores, each core has separate L1/L2 cache hierarchies
  - Port exists already on crossbar/SRI
  - L2 cache sizes expected to be 512KB or 1MB
- Shared integrated North-Bridge & Host-Bridge interface
  - Integrated memory controller & HyperTransport™ links route out same as today's implementation
  - Application-dependent resource contention…maybe 10% hit on average
- SSE3 support & some simple Instruction fixes



Existing AMD64 Processor Design

Socket/Package versus Core

- Physical chip/package plugs into a socket on the motherboard
- Populated socket contains a chip/package with a number of integrated cores.
- CPU Numbering scheme uses LSBs of Initial APIC ID to distinguish cores in one processor package.
  - High-order bits distinguish packages
  - Initial APIC ID provided by CPUID (eax=1)
- Example: 2 Packages populate 2 sockets and 2 cores per package/socket



"License by Sockets, Schedule by Cores"

- AMD suggests to developers that software be *licensed based on number of populated sockets, scheduled based on number of cores*.
  - For certain workloads, since N integrated cores share chip I/0 & resources, there could be less performance than N discrete cores
    - So why charge the same?
  - Focusing on sockets and not cores reduces end-user confusion
- So how can software distinguish sockets from cores and do the right thing?
- Unique initial APIC ID assigned to all cores
  - ACPI-MADT and MPS tables record all cores just like discrete case
- New extended CPUID function (eax=8000_0008) returns on any core the number of cores in the associated socket
    - Great for new software to use to figure number of cores
- But, legacy software doesn't know about new CPUID function. It understands only 2 models:
  - discrete SMP and SMT/hyperthreading …

"If it Quacks like a Duck …"

AMD

- So CPUID (eax=1) on all cores in a dual core pkg returns:
  - CPUID.HTT=1 (edx[28])
    - The fact that this may appear as hyperthreading to legacy software is just a happy co-incidence
  - CPUID.logical_number_of_processors = 2 (ebx[23:16])
  - New extended CPUID Feature bit, HTVALID, which tells New software if the HTT fields above report Hyperthreading
    - HTVALID will be zero on AMD dual core indicating no HTT support
- Legacy software support for 2-logical cores, while more restrictive, appears to work equally fine for 2-physical cores
  - Hyperthreading scheduling rules work fine for multi-core
  - Hyperthreading shared MSR rules work fine for multi-core
  - AMD evaluation of legacy software has thus far found no major problems with this assumption
- Migrating from hyperthreading rules to less restrictive multi-core rules becomes an optimization, not a requirement

More details for Break-out session

Various Instruction Set Additions and Corrections

SSE3

- Dual core is designed to provide 13 SSE Instructions reported by CPUID.SSE3 feature flag:
  - ADDSUB[PD,PS] xmm1, xmm2/m128
    - Provides interleaved packed add and subtract
  - FISTTP m16int/m32int/m64int
    - Like FISTP but with forced Truncation
  - HADD[PD,PS] xmm1, xmm2/m128
    - Horizontal Adds
  - HSUB[PD,PS] xmm1, xmm2/m128
    - Horizontal Subtracts
  - LDDQU xmm, m128
    - Special 128-bit Unaligned Load
  - MOV[DD,SHD,SLD]UP xmm1, xmm2/m64
    - Move and Duplicate some elements
- No Monitor or Mwait for Hyperthreading, which have separate CPUID flag anyway

More ISA Fixes & Additions

- LAHF and SAHF (load/store status flags in AH) now supported in 64-bit mode. Reported by CPUID.LAHF
- 64-bit Segment Limit Check Mechanism:
  - Assume segment-addressed access of form SEG:ADDR
  - if (64bit_mode && EFER[13] && (CPL > 0) &&
    (SEG==DS || SEG==ES || SEG==FS || SEG==SS))
    { limit = (SEG.G ? (SEG.limit << 12)+0xFFF : 0xFFF)));
    if (ADDR > ((0xFFFF << 32) + limit))
    generate_std_segment_limit_GP_fault();
    }
- Fast FXSAVE/FXRSTOR
  - If EFER.FFXSR==1, FXSAVE/FXRSTOR do not save/restore XMM0-XMM15 when executed in 64-bit mode at CPL 0.
  - Reported by CPUID.FFXSR

RDTSCP: Read Serialized TSC Pair

AMD

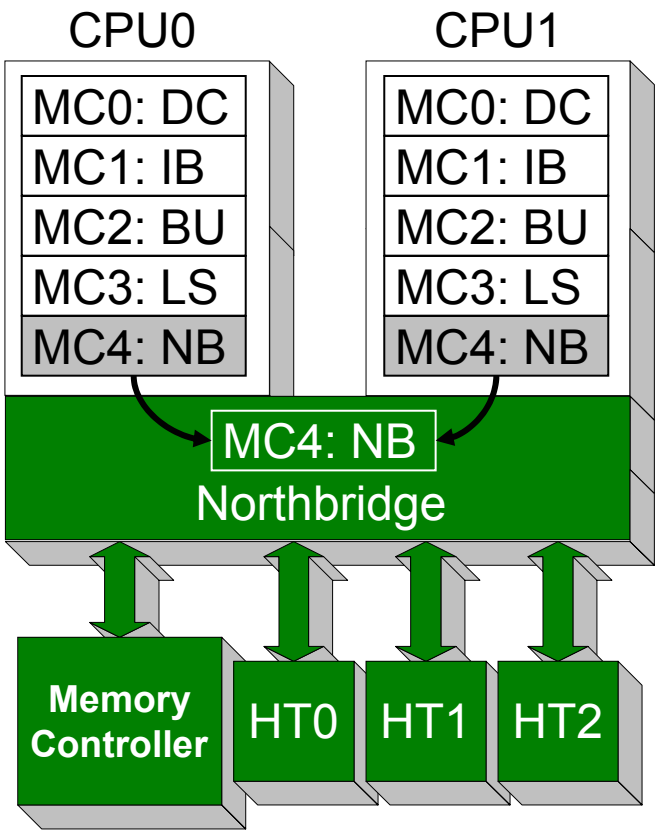
- New instruction, similar to RDTSC:
  - Returns 64-bit TSC value in %edx:%eax
  - Usage in non-ring-0 mode controlled by CR4.tsd
- But unlike RDTSC instruction:
  - Is a serializing operation -- prevents speculative reads of TSC
  - Returns TSC_AUX[31:0] MSR in %ecx at same time as TSC
    - OS initializes TSC_AUX to meaningful value
    - Atomicity ensures no context switch between read of TSC & TSC_AUX.
  - Availability determined by new extended CPUID feature flag
    - (CPUID(%eax=0x8000_0001)).edx[TBD]
- Allows TSC and OS-supplied value (such as CPU number) to be read atomically in a serializing way in user mode.
  - TSC rates between CPUs in MP-system may vary
  - Linux can use for user-mode get-time-of-day based on TSC
  - Linux can put CPU number in TSC_AUX so user-mode code knows which per-cpu adjustments to use to fix-up TSC value.

Dual Core: Shared NB Resource Software Issues

- If OS writes (or causes BIOS to write) any shared resource in the AMD K8™ Northbridge, that SW needs to be checked.
- SW may need to be changed to reflect the fact that such writes are global to both CPU0 and CPU1. Areas identified:
  - MCA NorthBridge MSRs
    - Possible Minor OS Issue, impact small
  - GART Programming Issues
    - Possible Minor OS Issue, impact small
  - CF8h/CFCh I/O Ports
    - Shared between both CPUs; no different than sharing in external host bridge
  - NB performance events
    - No OS impact; education in interpreting results.
- Bottom-line: these issues are minor, not show-stoppers from SW perspective.

MC4 Machine-Check MSR Shared between Cores

- MCG_CTL.NBE
  - read/write to NB register
- MC4_STATUS:
  - read/write to NB register
- MC4_ADDR:
  - read/write to NB register
- MC4_CTL:
  - read/write to NB register



- MCG_CTL.NBE
  - shared or dummy
- MC4_STATUS:
  - dummy
  - writes ignored
  - reads return 0.
- MC4_ADDR:
  - dummy
  - writes ignored
  - reads return 0.
- MC4_CTL:
  - shared or dummy

- Machine-check MC4 MSRs are shared between 2 cores.
- MCG_CAP.count on all cores = 5 (bank 0 to 4 exist).
- All NorthBridge (NB) Machine-checks (MC4 bank) are logged and raised in MC4 & MCG_STATUS MSRs of Core 0 only.

Dual Core Discussion

- SLIT/SRAT ACPI info
- Recommendations to Linux