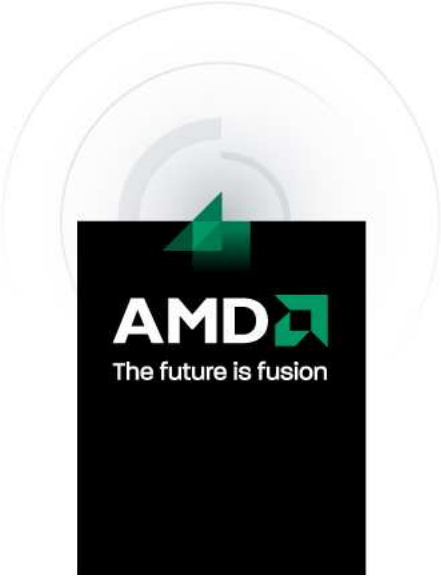
Quad Core AMD Opteron™ Processor Overview

Brian Waldecker, Ph.D.
Senior Member of Technical Staff
AMD, Austin
10/15/2008



Outline

AMD
Smarter Choice

1. Opteron and Multi-Core Architecture
2. Caches
3. Prefetching
4. Programming Hints
5. Performance Examples

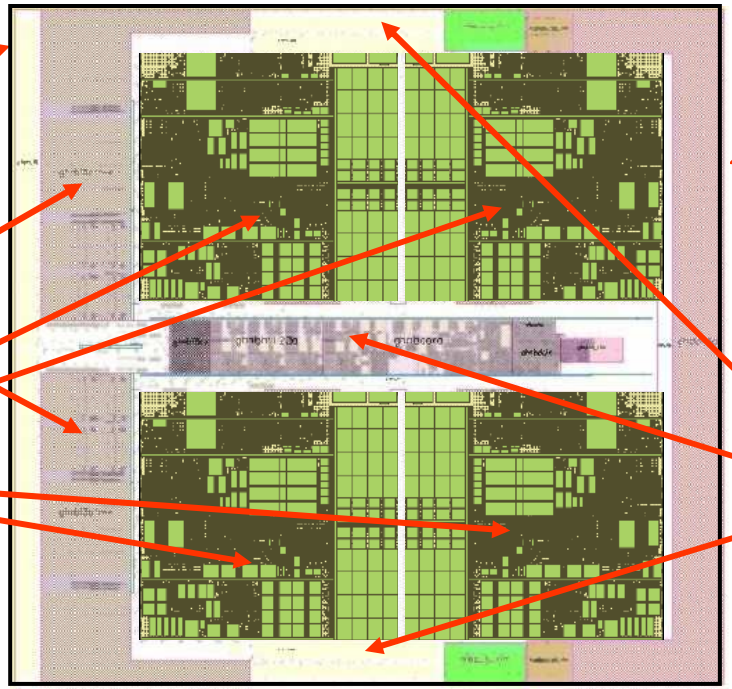
AMD's Third Generation Opteron™ Processor Technology

AMD Smarter Choice

Native quad core die

Expandable shared L3 cache

IPC enhanced CPU cores



Optimized for 65nm SOI and beyond

Enhanced Direct Connect Architecture and Northbridge

- 32B instruction fetch
- Improved branch prediction
- Out-of-order load execution
- Up to 4 DP FLOPS/cycle
- Dual 128-bit SSE dataflow
- Dual 128-bit loads per cycle
- Instruction Extensions
- Power Management Extensions

- Enhanced HyperTransport™ links
- Enhanced crossbar
- DDR2 with migration path to DDR3
- Enhanced power management and RAS

Comprehensive Upgrades for SSE128

128bit FPU

AMD Smarter Choice

Parameter	2nd Gen. Opteron	3rd Gen. "Barcelona"
SSE Exec Width	64	128 + SSE MOVs
Instruction Fetch Bandwidth	16 bytes/cycle	32 bytes/cycle + unaligned Ld-Ops
Data Cache Bandwidth	2 x 64bit loads/cycle	2 x 128bit loads/cycle
L2/NB Bandwidth	64 bits/cycle	128 bits/cycle
FP Scheduler Depth	36 Dedicated x 64-bit ops	36 Dedicated x 128-bit ops

Can perform SSE MOVs in the FP "store" pipe

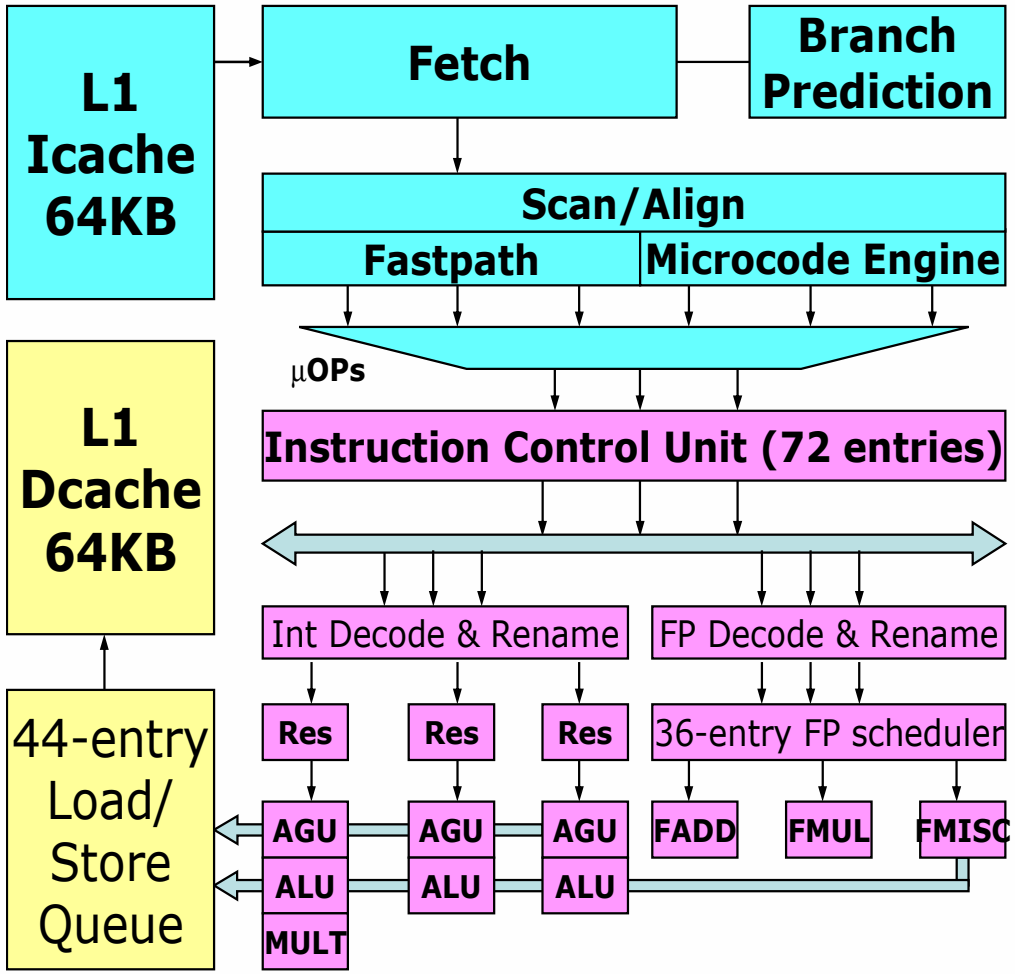
- Execute two generic SSE ops + SSE MOV each cycle (+ two 128-bit SSE loads)

SSE Unaligned Load-Execute mode

- Remove alignment requirements for SSE ld-op instructions
- Eliminate awkward pairs of separate load and compute instructions
- *To improve instruction packing and decoding efficiency*

Core IPC improvements

- Improve Branch Prediction.
- TLB enhancements.
- More out of order Ld/St capability.
- New Instructions
  - POPCNT / LZCNT
  - EXTRQ / INSERTQ
  - MOVNTSD / MOVNTSS
- Fastpath support for FP to Integer data movement.



TLB Enhancements

AMD
Smarter Choice

- Support for 1GB pagesize (4k, 2M, 1G)
- 48 bit physical addresses = 256TB (increase from previous 40bits)
- Data TLB
  - L1 Data TLB
    - 48 entries, fully associative
    - all 48 entries support any pagesize
  - L2 TLB
    - 512 4k entries, or
    - 128 2M entries
- Instruction TLB
  - L1 Instruction TLB
    - fully associative
    - support for 4k or 2M pagesizes
  - L2 Instruction TLB

Data Prefetch

AMD
Smarter Choice

- **Hardware prefetching**
  - DRAM prefetcher
    - tracks positive, negative, non-unit strides.
    - dedicated buffer (in NB) to hold prefetched data.
    - Aggressively use idle DRAM cycles.
  - Core prefetchers
    - Does hardware prefetching into L1 Dcache.
- **Software prefetching instructions**
  - MOV (prefetch via load / store)
  - prefetcht0, prefetcht1, prefetcht2 (currently all treated the same)
  - prefetchw = prefetch with intent to modify
  - prefetchnta = prefetch non-temporal (favor for replacement)

Cache Hierarchy

AMD Smarter Choice

Dedicated L1 cache

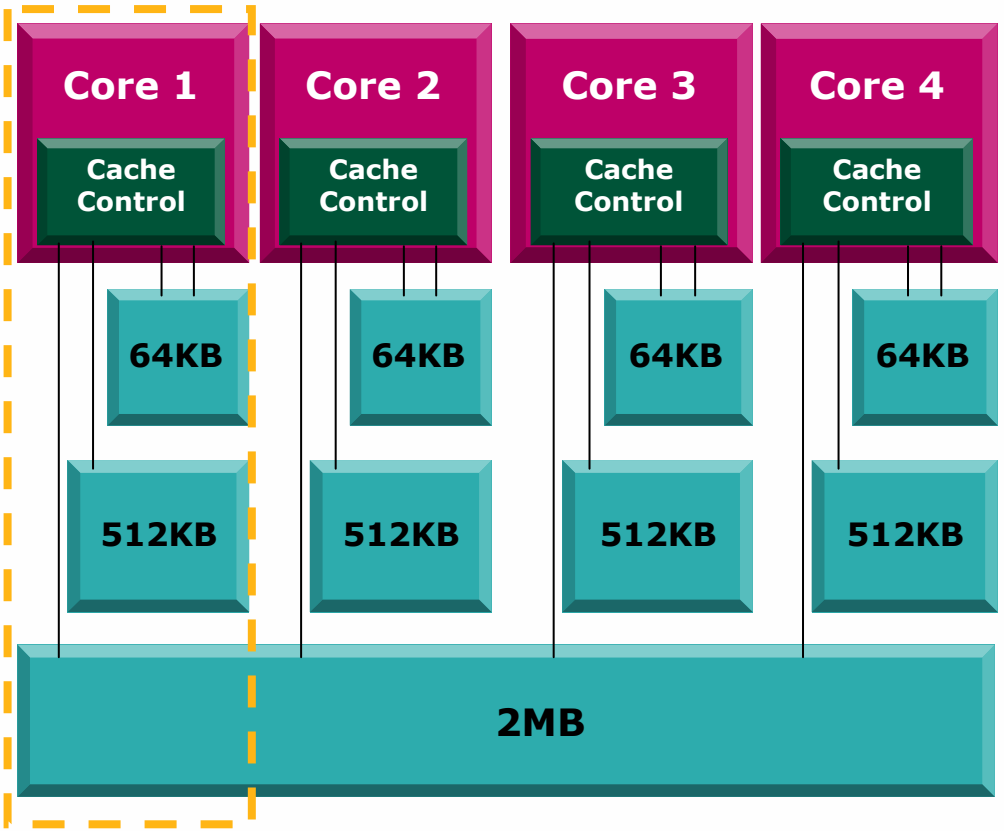
- 2 way associativity.
- 8 banks.
- 2 128bit loads per cycle.

Dedicated L2 cache

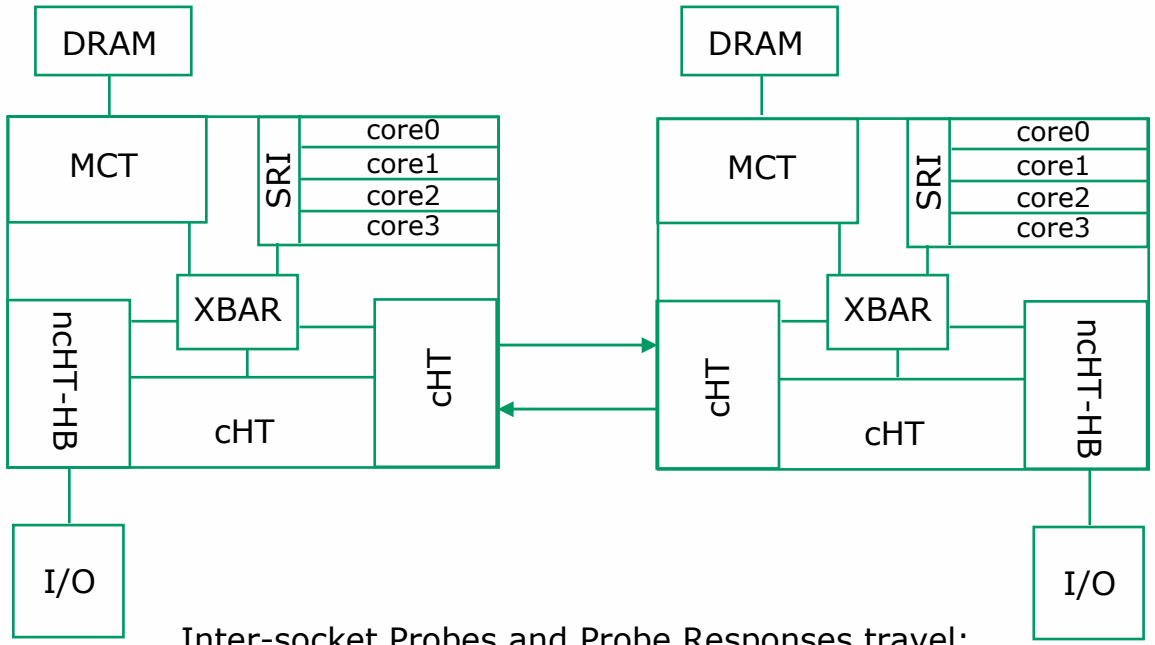
- 16 way associativity.

Shared L3 cache

- 32 way associativity.
- fills from L3 leave likely shared lines in L3.
- sharing aware replacement policy.



Multi-socket System overview

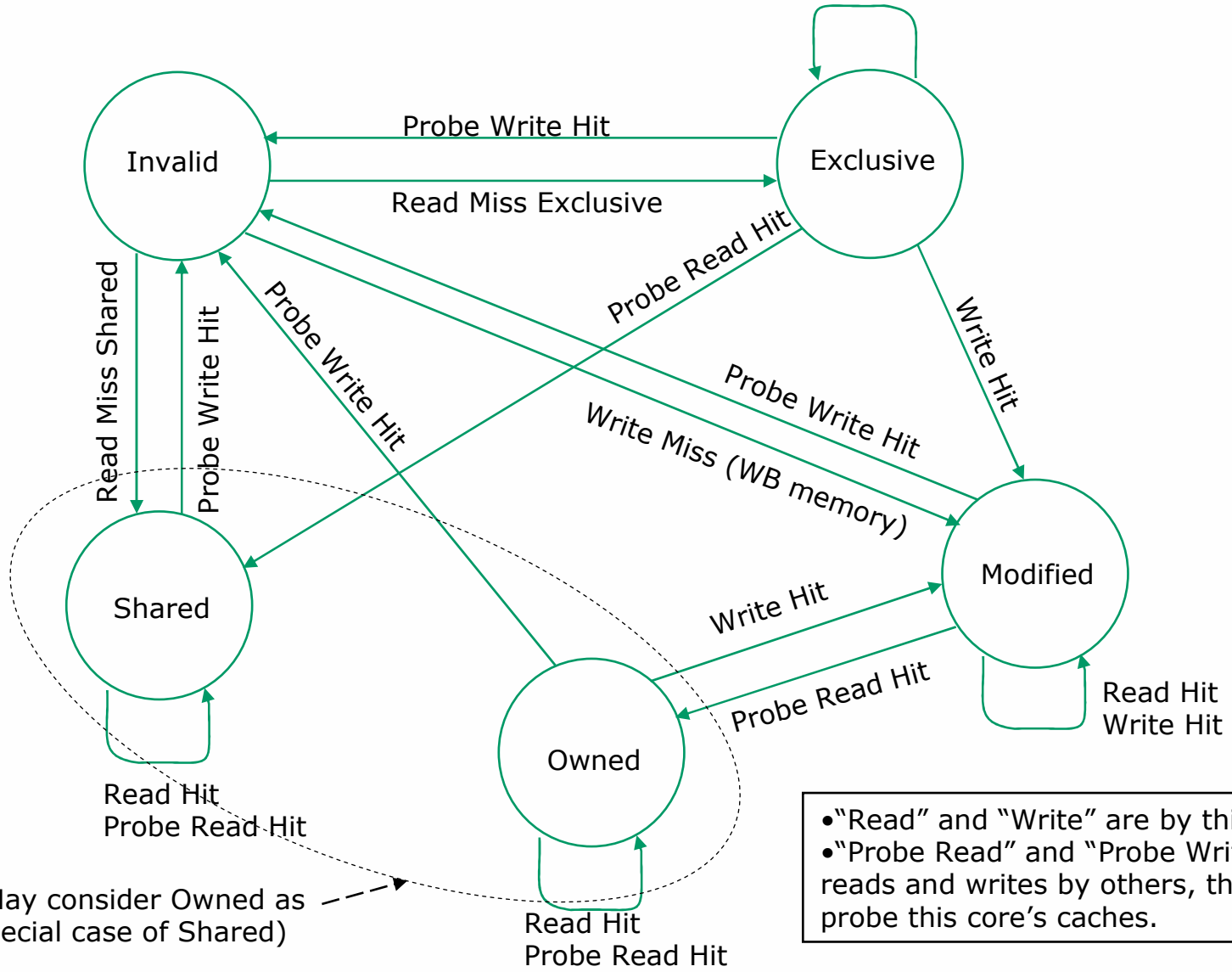


Inter-socket Probes and Probe Responses travel:
SRI -> XBAR -> cHT -> cHT -> XBAR -> SRI

Probes Requests initiate at home memory node, but return directly to node making initial memory request.

key:
cHT = coherent HyperTransport
ncHT = non-coherent HyperTransport
XBAR = crossbar switch
SRI = system request interface (memory access, cache probes, etc.)
MCT = memory controller
HB = host bridge (e.g. HT to PCI, SeaStar, etc.)

Cache Coherency (MOESI protocol)



- "Read" and "Write" are by this core.
- "Probe Read" and "Probe Write" are reads and writes by others, that must probe this core's caches.

Cache Coherency (Practical Advice)

AMD
Smarter Choice

- Avoid shared read and shared write data in same cacheline.
- Avoid gratuitously modifying shared data.
  - Sharing aware L3 helps within a chip, but doesn't make such updates free.
  - Minimize false sharing where compiler has to play it safe.
- Requirement to wait for all probe responses means local memory and remote cache accesses have similar latencies.
  - Sometimes thinking of just memory is just fine.
  - Let library and compiler writers worry about being uber-clever.
- Aliasing, Aliasing, Aliasing of addresses. (Help the compiler).
  - If compiler's unsure about potential aliasing it must play it safe and generate extra stores and loads, instead of working only with registers.

Sharing Aware (Partially Inclusive) L3

AMD Smarter Choice

- Inclusive vs. Exclusive Cache Paradigms
  - Inclusive: L3 contains L2 contains L1 (i.e. supersets).
  - Exclusive: L3, L2, and L1 are disjoint sets.
- L3 tracks core that last touched cacheline.
  - Read request from a different core cause L3 to retain copy of data in O or S state.
    - i.e. assumes data shared, hence inclusive behavior.
  - Read from same core causes L3 to return data and invalidate cacheline in L3 (not retain a copy).
    - i.e. assumes data not shared, hence exclusive cache behavior.
  - Writes from same core or different cores implemented according to exclusive cache paradigm.

A Few Programming Hints
L3 Cache Data Sharing

AMD
Smarter Choice

Optimizing Producer Consumer ring buffers

1. Data gets into L3 by spilling from L1+L2.
2. Consumer must "lag" producer by at least L1+L2 bytes so data is in L3 when consumed. (artifact of victim cache design).
3. Producer must not get too far ahead or it will flood L3 and evict unconsumed lines.
4. Producer must lag consumer by at least L1+L2 bytes so data is written into L3 with minimal collision with Consumer. (MOESI artifact).
5. Consumer should use PrefetchW* to prefetch data.
6. Ring Buffer should be at least 2 x (L1+L2) in size, plus some cushion.
7. Remember, to account for L3 size per Producer-Consumer pair. (when running 1 pair vs. 2 pairs per chip).

* PrefetchW brings data into cache in M state. Otherwise data might come in as S state and L3 retains a copy in O state.

A Few Programming Hints

AMD
Smarter Choice

• Use SSE2 instructions that modify entire 128bit SSE register instead of preserving one half.

•Generally good to prefetch 6 to 8 cachelines ahead

- Latency-Bandwidth product estimates how much data must be "in-flight"
  - 1P, DDR2-800 ~= 53ns * 10GB/s = 530 Bytes = ~8 cache lines in flight.
  - 2P, DDR2-667 ~= 81ns * 17GB/s = 1377 Bytes = ~21 cache lines in flight (combined across both Northbridges).

• Try to have 100 cycles of computation in loop body between successive prefetches

• Avoid issuing multiple software prefetches to the same cacheline

• Unroll loops enough times so each iteration works on 1 or more cachelines of data.

note: neither hw or sw prefetches will be allowed to generate page faults, but a TLB miss on a prefetch can initiate a TLB fill.

A Few Programming Hints

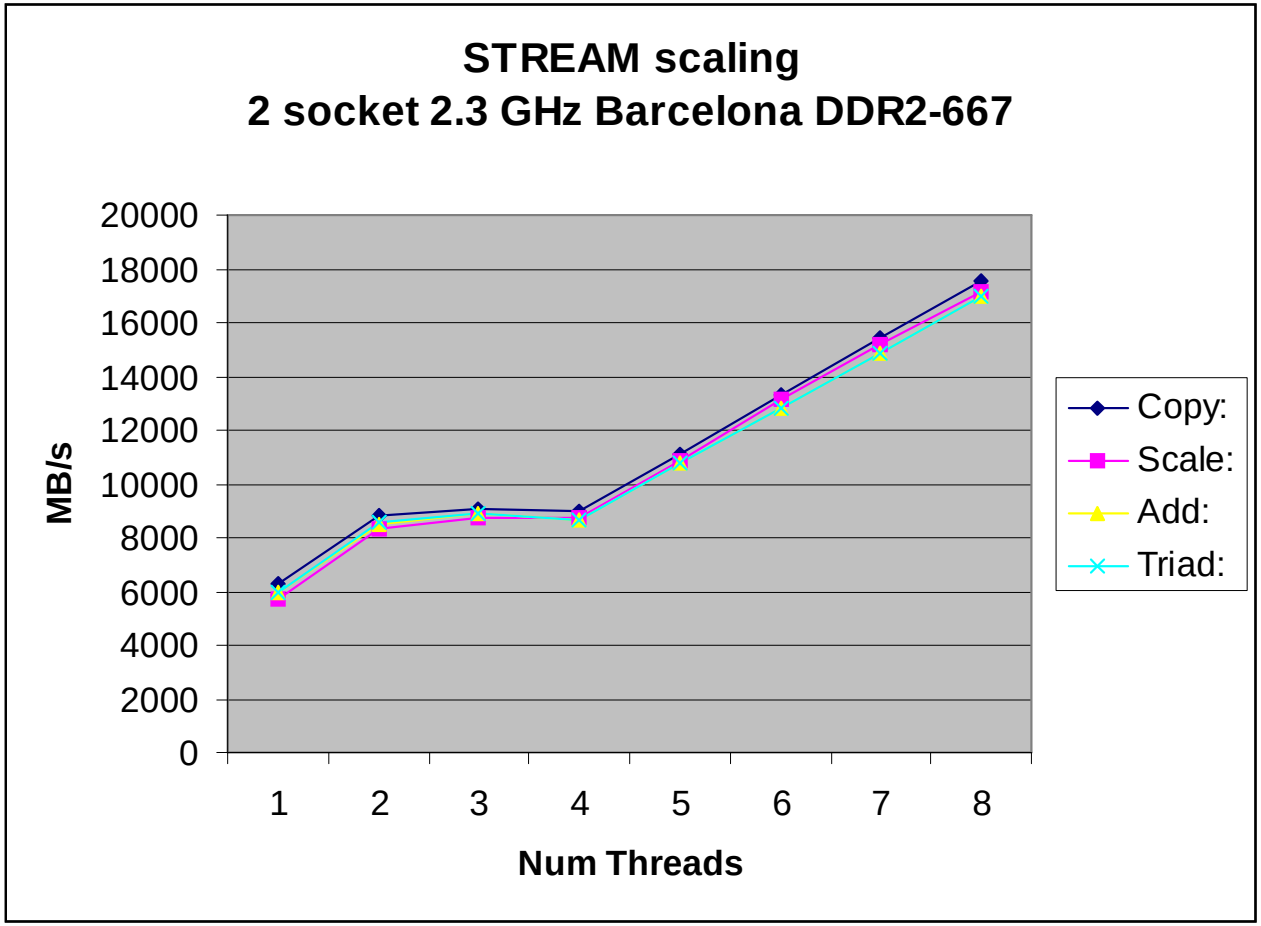
AMD Smarter Choice

Which Prefetch to use ?

Data	Less than ½ L1 size	Less than ½ L2 size or of unknown size		Greater than ½ L2 size
		Reused	Not Reused	
Read only	`prefetch or prefetchnta`	`prefetch`	`prefetchnta`	`prefetchnta`
Sequential read only	`hwprefetcher + prefetch`	`hwprefetcher + prefetch`	`prefetchnta`	`prefetchnta`
Read-write	`prefetchw`	`prefetchw`	`prefetchnta`	`prefetchnta`
Sequential read-write	`prefetchw`	`prefetchw`	`prefetchnta`	`prefetchnta`
Write only	`prefetchw`	`prefetchw`	`movnt`	`movnt`
Sequential write only	`hwprefetcher + prefetchw`	`hwprefetcher + prefetchw`	`movnt`	`movnt`

SOME MULTI-CORE PERFORMANCE EXAMPLES

OpenMP Stream – Two Sockets



Supermicro H8DMU serverboard,
Two 2.3GHz Barcelona cpus,
8 x 2GB DDR2-667 CL5 memory
SLES10 SP1, Pathscale 3.1 C compiler

Levesque Loops 41080 and 41081

AMD Smarter Choice

```
#define LCOUNT 1000000
#define N 11
#define AIDX2 128
#define BIDX2 14

double A[N][AIDX2];
double B[N][BIDX2];
double c0, c1, c2, c3, c4, c5, c6, c7, c8;

int Work41080() {
  int i,j,l;

  for (l=0; l < LCOUNT; l++) {
    for (i=0; i < N; i++) {
      A[i][0] = c1*A[i][12] + c2*A[i][11] + c3*A[i][10] +
                c4*A[i][9]  + c5*A[i][8]  + c6*A[i][7]  +
                c7*A[i][6]  + c0*A[i][4]  + A[i][5]    + A[i][2];
    }
  }
  return(0);
}

int Work41081() {
  int i,j,l;

  for (l=0; l < LCOUNT; l++) {
    for (i=0; i < N; i++) {
      B[i][0] = c1*B[i][12] + c2*B[i][11] + c3*B[i][10] +
                c4*B[i][9]  + c5*B[i][8]  + c6*B[i][7]  +
                c7*B[i][6]  + c0*B[i][4]  + B[i][5]    + B[i][2];
    }
  }
  return(0);
}
```

Notes:
- C translation of original Fortran code to facilitate performance counter instrumentation.
- Array indices were reversed accordingly.

IPC, Cycle, and Instruction Counts

AMD Smarter Choice

- Instrumentation inserted just before and after loops to read HW performance counters.
- Only difference in loops was dimension of the fast moving array index.
  - Each iteration on A[i] strides 1024 bytes
  - Each iteration on B[i] strides 112 bytes
- Powers of two alignment can sometimes be suboptimal, but not always.
- Best Advice: Proof is in the pudding, try padding and see.

Loop	TSC	User Instructions	User Cycles	IPC
41080 (A[i][])	195183582	323016777	195017197	1.656350
41081 (B[i][])	198239940	323016777	198048238	1.631001

A[11][128], sizeof(A[0][])=1024, (&A[i+1][0] - &A[i][0])=1024 bytes
B[11][14], sizeof(B[0][])=112, (&B[i+1][0] - &B[i][0])=112 bytes

Multi-core Performance Example

SPEC OMPL2001 (SPEC-HPG OpenMP benchmark)
313.swim_l (shallow water ocean model)

Opteron™ (Barcelona) System

Tyan Thunder n425QE (S4985E)
Four Opteron 8356 CPUs @ 2.3GHz
16 x 2GB DDR2-667
SLES10 SP1 X86_64
PathScale Compiler Suite 3.1

Multi-core Performance Example

313.swim_l (shallow water modeling, large dataset)

AMD Smarter Choice

Three sets of compiler* flags used:

"Ofast"
-mp -Ofast -mcpu=barcelona -OPT:early_mp=on -mcmodel=medium

"Ofast_simd0"
-mp -Ofast -mcpu=barcelona -OPT:early_mp=on **-LNO:simd=0** -mcmodel=medium

"Ofast_movnti2500"
-mp -Ofast -mcpu=barcelona -OPT:early_mp=on **-CG:movnti=2500** -mcmodel=medium

Flags	Runtime in secs. (mins.)
Ofast	7194s (120m)
Ofast_simd0	1736s (29m)
Ofast_movnti2500	1785s (30m)

Too much of a good thing? (streaming stores or vectorization)

* Pathscale ™ Compiler Suite, Version 3.1, SLES10 SP1

Multi-core Performance Example

313.swim_l (shallow water modeling, large dataset)

AMD Smarter Choice

Profiling shows

Problem Size:
7701 x 7701 grid, REAL*8
452MB per array (5.8GB total)

Ofast

samples	%	symbol name
598797565	79.1052	__ompdo_calc3_1
90931114	12.0126	__ompregion_calc2_1
48912887	6.4617	__ompregion_calc1_1
17952271	2.3716	__ompdo_MAIN__1
153331	0.0203	__ompdo_calc3z_1

Program Structure:
10 NCYCLE=NCYCLE+1
  Calc1 (writes 4 arrays)
    ...
  Calc2 (writes 3 arrays)
    ...
  Calc3 (writes 6 arrays) ← Thrashing WCBufs!
    ...
GOTO 10

Ofast_simd0

samples	%	symbol name
68233066	34.4688	__ompregion_calc2_1
61859772	31.2492	__ompdo_calc3_1
48931003	24.7181	__ompregion_calc1_1
18617176	9.4047	__ompdo_MAIN__1
132326	0.0668	__ompdo_calc3_2

Ofast_movnti2500

samples	%	symbol name
68314854	34.4881	__ompregion_calc2_1
61749164	31.1735	__ompdo_calc3_1
48932509	24.7031	__ompregion_calc1_1
18770527	9.4761	__ompdo_MAIN__1
132286	0.0668	__ompdo_calc3_2

Oprofile: Counted CPU_CLK_UNHALTED events (Cycles outside of halt state)

Multi-core Performance Example

313.swim_l (shallow water modeling, large dataset)

AMD Smarter Choice

Performance Counters

Ofast = default
Ofast_simd0 = -LNO:simd=0
Ofast_movnti2500 = -CG:movnti=2500

		default	"-LNO:simd=0"	"-CG:movnti=2500"
	CPI	14	2.55	3.7
	Clocks(B)	15373	4107	4146
	Insts(B)	1100	1618	1096
	L3Req	440.000	161.800	142.480
	L3Miss	330.000	134.294	134.808
	totSSE	792.000	1164.960	789.120
	FPadd pipe	583.000	388.320	252.080
absolute(B)	FPmult pipe	242.000	210.340	151.248
	FPstore pipe	291.500	142.384	113.984
	PgOpen	253.000	37.214	36.168
	PgClose	233.200	114.878	111.792
	PgCflct	114.400	63.102	61.376

* Pathscale ™ Compiler Suite, Version 3.1, SLES10 SP1

Summary

AMD
Smarter Choice

Core Improvements:

- Core IPC, Caches and TLB.
- FPU and memory performance.

Multi-Core Implications:

- Flop rich environment.
- Shared L3 data, “partially inclusive” nature.

Multi-Socket Implications (looking forward – i.e. XT5):

- NUMA awareness.
- Cache Probing, Data Transfer across HT links.

References

- AMD Opteron Processor Families Technical Documentation
  - http://www.amd.com/us-en/Processors/TechnicalResources/0,,30_182_739_9003,00.html
- *Bios and Kernel Developers Guide* (BKDG) on AMD website.
  - RevF cpus are denoted as “Family 0Fh”.
  - Quadcore BKDG is “Family 10h”.
    - Portions useful to others than just Bios and Kernel developers.
- *Software Optimization Guide for AMD64 Processors*
  - Quadcore version available on AMD website.

Trademark Attribution