

“Understanding Bulldozer architecture through Linpack benchmark”

By Joshua.Mora@amd.com

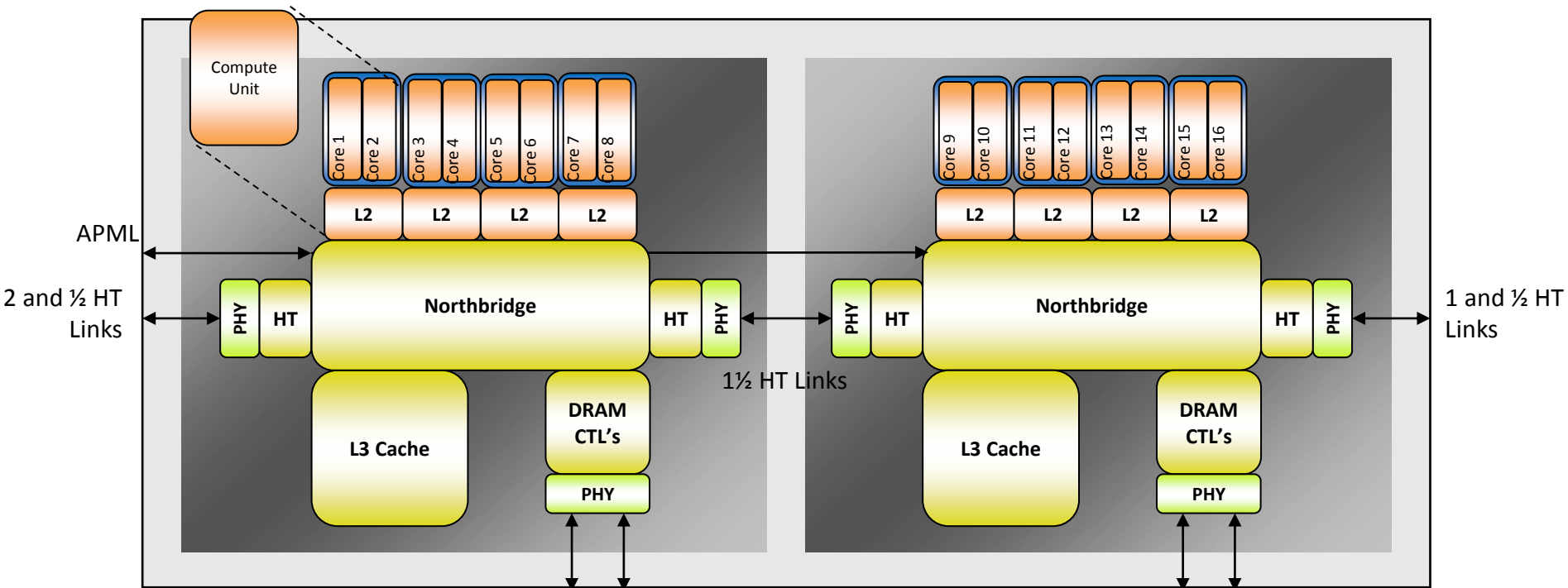
Abstract: AMD has recently introduced a new core architecture named Bulldozer in the newly released multicore processors such as Interlagos processor. We will cover experimentally through Linpack benchmark some of the key features of this new processor such as the shared floating point unit on the Bulldozer core, the fused multiply add instructions (FMA4) and power management that allows cores to boost. Emphasis on the appropriate software ecosystem such as optimized libraries (ACML), compiler flags (open64) and operating system will be discussed as well so you can fully exploit the new generation of AMD processors.

Agenda

- G34 socket, BD module
- Individual and shared resources, Caches, FPU
- DP floating point calculations: SSE2 vs FMA4
- Real FLOPs/clock per BD module (efficiency)
- SW ecosystem: OS, O64, ACML, Profilers
- Testing BD FPU architecture with DGEMM
- Multithreaded runs 1-2 threads/BD unit, 1-4BD units (8 threads)
- Testing BD FPU architecture with HPL
- CodeAnalyst session on HPL
- Advanced Power Management (ie. boost)
- HPCMODE (no need to disable APM for HPL)
- HPL benchmark data on Interlagos processor
- Detecting Thermal Throttling when running HPL

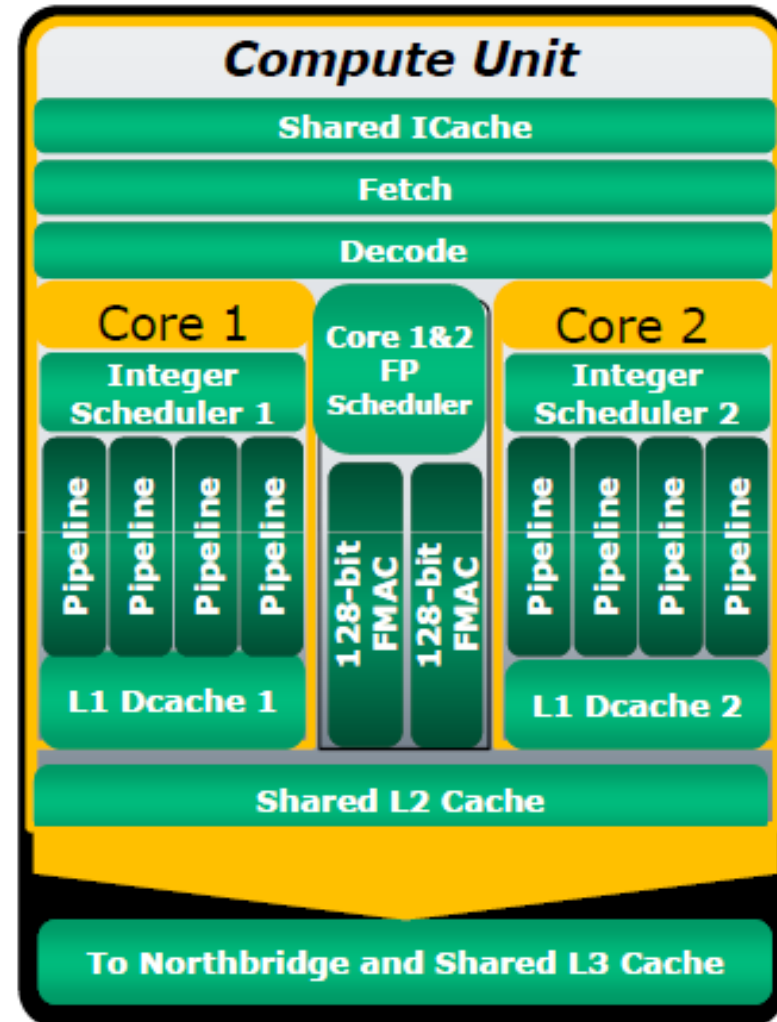
G34 socket, BD module

Example of Interlagos processor, with 16 cores, 2.3GHz in G34 socket.
 4+4 Bulldozer modules on 2 numanodes connected through coherent
 HyperTransport. Each numanode has 2 memory channels.
 Delivers 18.5 GB/s x 2, 60 DP GF/s x2 under 130W (115W TDP).



Individual and shared resources

- HPC workloads are using all the cores for the same nature of computation, mostly synchronized.
- High workload flexibility such as in Cloud under power budget.
- Example: Cloud workloads can use 1 core for integer work and the other the whole FPU for number crunching



Cache hierarchy, how data flows

Shared L1I

64 KB, 2-way, 64 B line, 32 B fetch / clk

Dedicated Per Core L1D

16 KB, 4-way, 64 B line, Write Through
2 x 128 bit LD / clk , 1 x 128 bit ST / clk
4 clk LD to use latency
40 entry LD queue, 24 entry ST queue

Shared L2

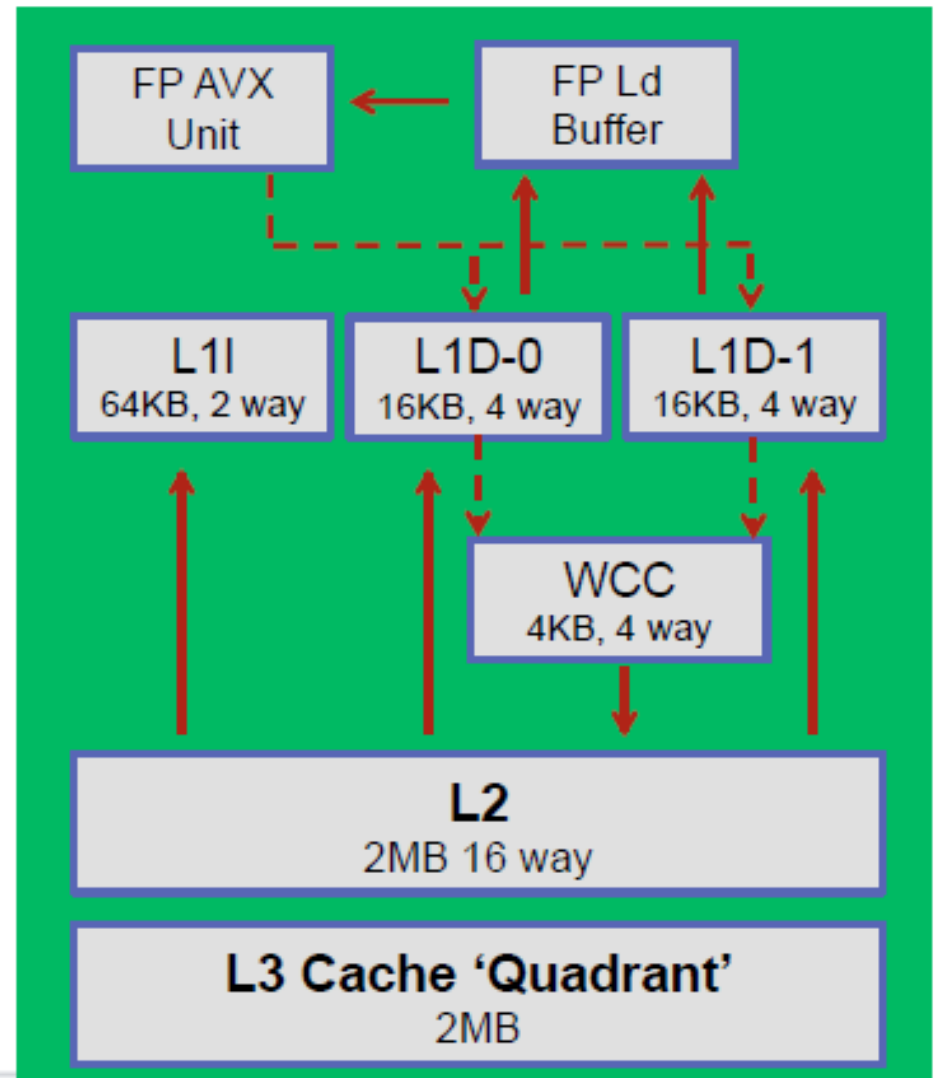
Unified 2MB, 16-way, 64 B line
20 clock LD to use latency
Up to 23 outstanding misses

WCC

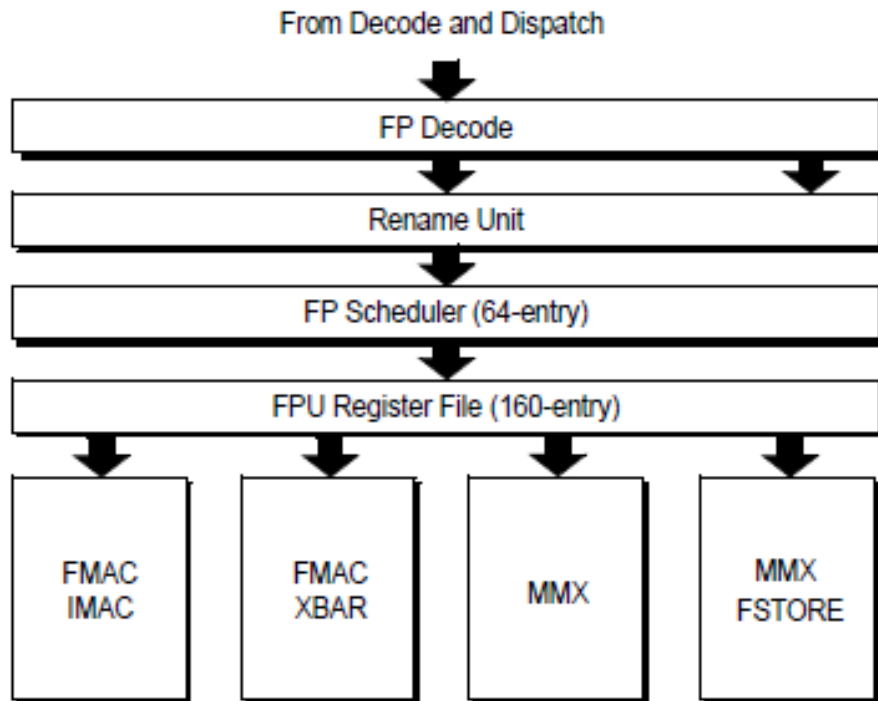
4 KB, 4 way, coalesces stores

Distributed L3

4 x 2MB quadrants
Victim cache for L2, non-inclusive



Focusing into the FPU of BD



Tens of “operations in flight” through the FP scheduler

SSE2 and FMA4 instructions are executed in pipes 0 and 1

Ex. SSE2: **ADDPD, MULPD**

Ex. FMA4: **VFMADDPD**

Table 8. Mapping of Pipes to Floating-Point Units

Pipe 0	Pipe 1	Pipe 2	Pipe 3
FPFMA fmul, fadd, fmac	FPFMA fmul, fadd, fmac	FPMAL avx, simd, mmx, ALU	FPMAL avx, simd, mmx, ALU
FPCVT fconverts	FPXBR shuffles, packs, permutes		FPSTO fpstore
FPMMA avx, simd, mmx, multiplier			

FMA instruction latencies on FMA0/1 pipes

VEXTRACTPS_128_reg	XBR[P1]/STO[P3]	FastPath Double	2/2
VFMADDPD_256_reg	FMA[P0 P1]	FastPath Double	6

272

Instruction Latencies

Appendix B



47414 Rev. 3.06 January 2012

Software Optimization Guide for AMD Family 15h Processors

Table 12: FPU Instruction Latencies (Continued)

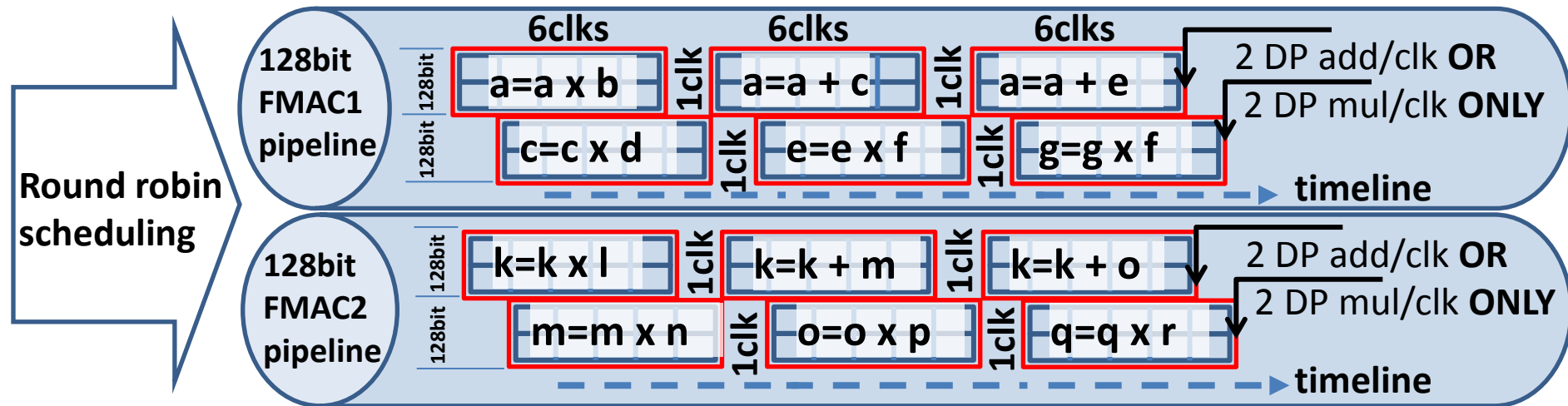
Instruction	Pipes	Decode Type	Latencies
VFMADDPD_128_reg	FMA[P0 P1]	FastPath Single	6
VFMADDPD_256_reg	FMA[P0 P1]	FastPath Double	6
VFMADDS_128_reg	FMA[P0 P1]	FastPath Single	6
VFMADDS_256_reg	FMA[P0 P1]	FastPath Single	6
VFMADDSUBPD_128_reg	FMA[P0 P1]	FastPath Single	6
VFMADDSUBPD_256_reg	FMA[P0 P1]	FastPath Double	6
VFMADDSUBPS_128_reg	FMA[P0 P1]	FastPath Single	6
VFMADDSUBPS_256_reg	FMA[P0 P1]	FastPath Double	6
VFMADDSUBPD_128_reg	FMA[P0 P1]	FastPath Single	6
VFMADDSUBPD_256_reg	FMA[P0 P1]	FastPath Double	6

From SWOG Family 15h

DP floating point calculations: SSE2

$$\left. \begin{aligned} \text{FMAC1 pipeline} &= (\dots(((axb + cxd) + exf) + gfx) \dots) \\ \text{FMAC2 pipeline} &= (\dots(((kxl + mxn) + oxp) + qxr) \dots) \end{aligned} \right\} \text{Done in parallel at each FMAC}$$

SSE2 instruction execution on 2 x 128bit FMAC units (4 DP F/clk/BD)

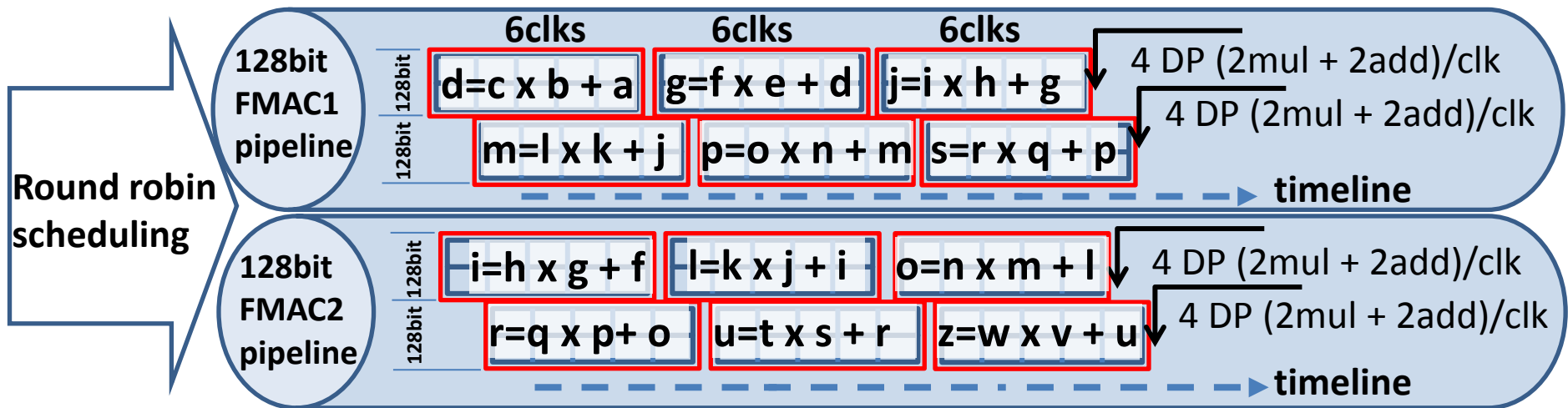


At each clock you can have many adds or multiplies in flight (6 clocks latency per operation) per pipeline (just pictured 2 + 1clock overhead). It takes 13 clocks to do 1 multiply + 1 add (eg. $axb + cxd$) with SSE2. It can only crunch 2 DP FLOP per clock per pipeline: (4 DP F/clock/BD module)

DP floating point calculations: FMA4

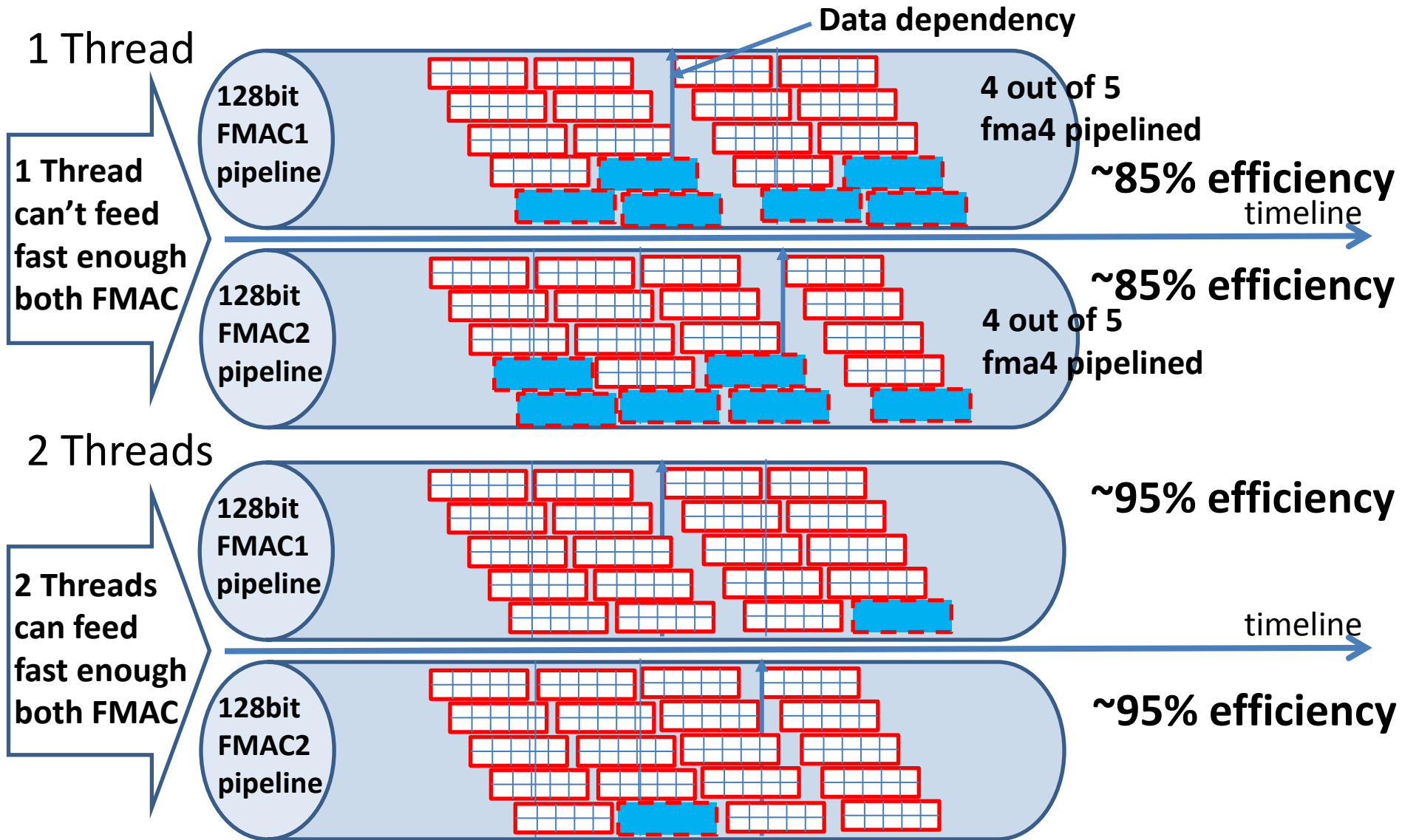
$$\begin{array}{l}
 \text{FMAC1 pipeline} = (...((cxb + a) + fxe) + ixh)... \\
 \text{FMAC1 pipeline} = (...((ljk + j) + oxn) + rxq)... \\
 \text{FMAC2 pipeline} = (...((hgx + f) + kxj) + nxm)... \\
 \text{FMAC2 pipeline} = (...((qxp + o) + txs) + wxv)...
 \end{array}
 \left. \vphantom{\begin{array}{l} \text{FMAC1 pipeline} \\ \text{FMAC1 pipeline} \\ \text{FMAC2 pipeline} \\ \text{FMAC2 pipeline} \end{array}} \right\} \text{Done in parallel at each FMAC}$$

FMA4 instruction execution on 2 x 128bit FMAC units (8 DP F/clk/BD)



At each clock you can have many fused multiply-adds in flight (6 clocks latency per operation) per pipeline (just pictured 2 + 0 clock overhead). It takes 6 clocks to do 2 fused multiply-adds (eg. $d=cxb+a$) with FMA4. It can crunch 4 DP FLOP per clock per pipeline: (8 DP F/clk/BD module)

Real FLOPs/clock per BD module (efficiency)



SW ecosystem: OS, O64, ACML, Profilers

- OS with Interlagos/Bulldozer support:
 - RHEL6.2, SLES11sp2, W2K8R2 and their derivatives (eg. CentOS, Scientific Linux, Ubuntu..)
- Compiler with Interlagos/Bulldozer support:
 - Open64 , GCC (-march=bdver1), PGI (-tp bulldozer-64)
- Math libraries with Interlagos/Bulldozer support
 - ACML, AMDlibm
- Profiler with Interlagos/Bulldozer support
 - CodeAnalyst, Oprofile, PAPI, perf
- Further info provided on last slide

Testing BD FPU architecture with DGEMM

Understanding DGEMM with a bit of algebra

From BLAS:

DGEMM performs a matrix-matrix operations

$C := \alpha * \text{op}(A) * \text{op}(B) + \beta * C$, in double precision (64bit)

$\text{op} = \{\text{No Transpose}, \text{Transpose}\}$, α and β are scalars

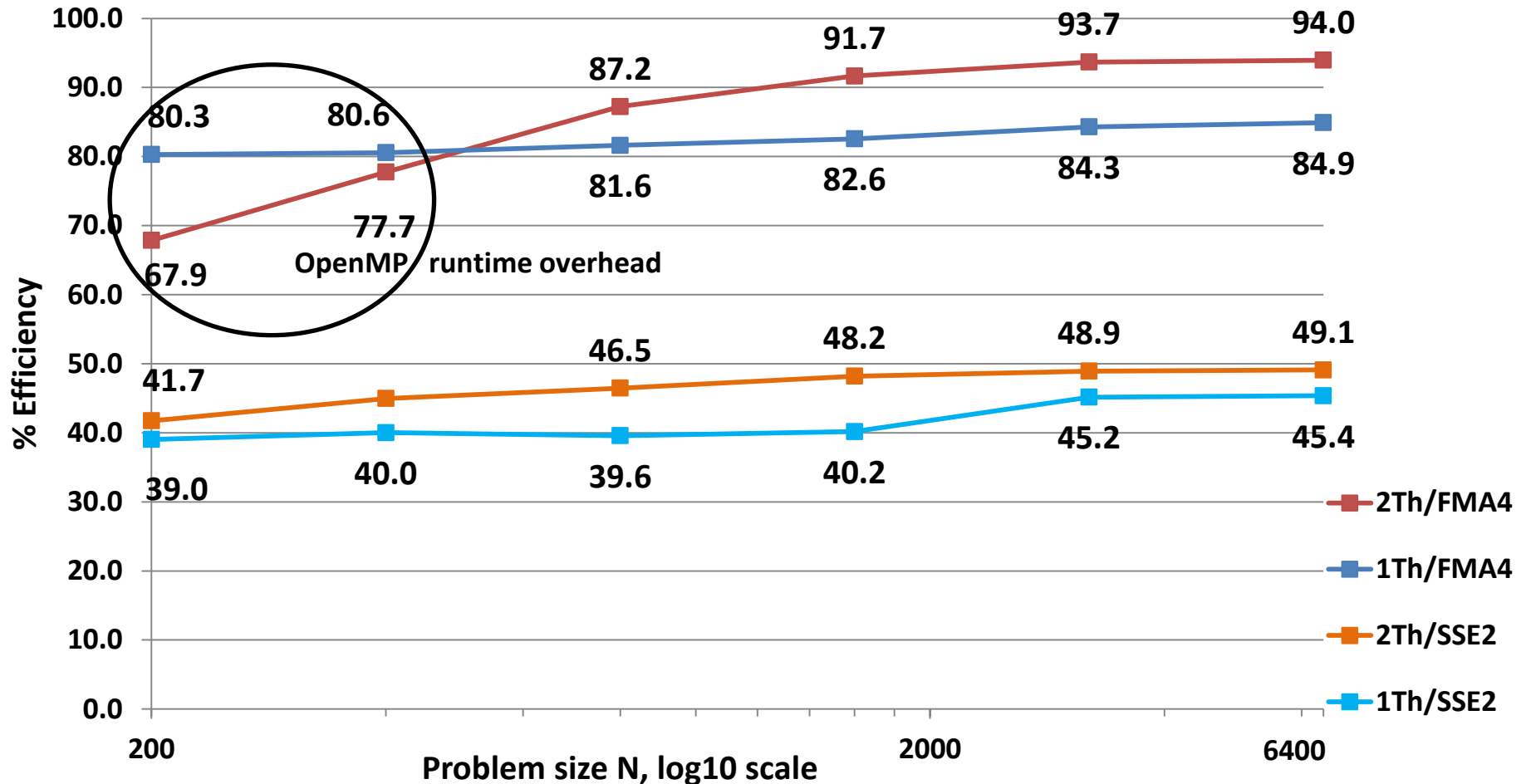
Expressed as inner products of rows of A (a) by columns of B (b):

$$\begin{aligned}
 \mathbf{A} * \mathbf{B} &= \begin{matrix} \mathbf{a} \\ \left(\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array} \right) \end{matrix} \begin{matrix} \mathbf{b} \\ \left(\begin{array}{cc} a & d \\ b & e \\ c & f \end{array} \right) \end{matrix} = \begin{matrix} \mathbf{a} \cdot \mathbf{b} \\ \left(\begin{array}{cc} 1a + 2b + 3c & 1d + 2e + 3f \\ 4a + 5b + 6c & 4d + 5e + 6f \\ 7a + 8b + 9c & 7d + 8e + 9f \end{array} \right) \end{matrix} \\
 \mathbf{a} \cdot \mathbf{b} &= (a_1 \quad a_2 \quad \dots \quad a_n) \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \underbrace{a_1 b_1 + a_2 b_2 + \dots + a_n b_n}_{\sum_{i=1}^n a_i b_i}
 \end{aligned}$$

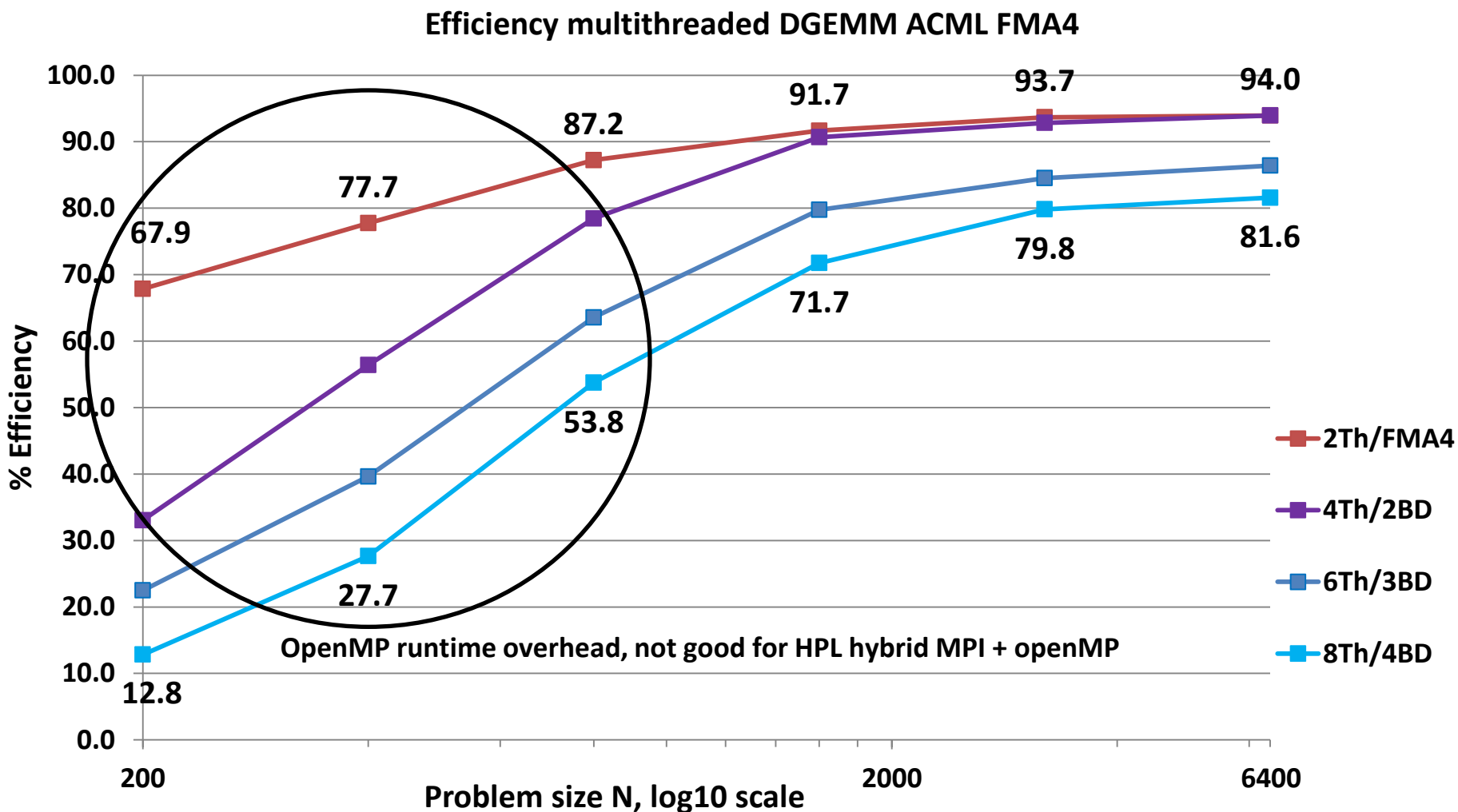
Lots of multiply adds, can be expressed as FMA4

Multithreaded runs 1-2 threads/BD unit

Efficiency BD unit, multithreaded DGEMM ACML FMA4 vs SSE2



Multithreaded runs 1-4 BD units (8threads)

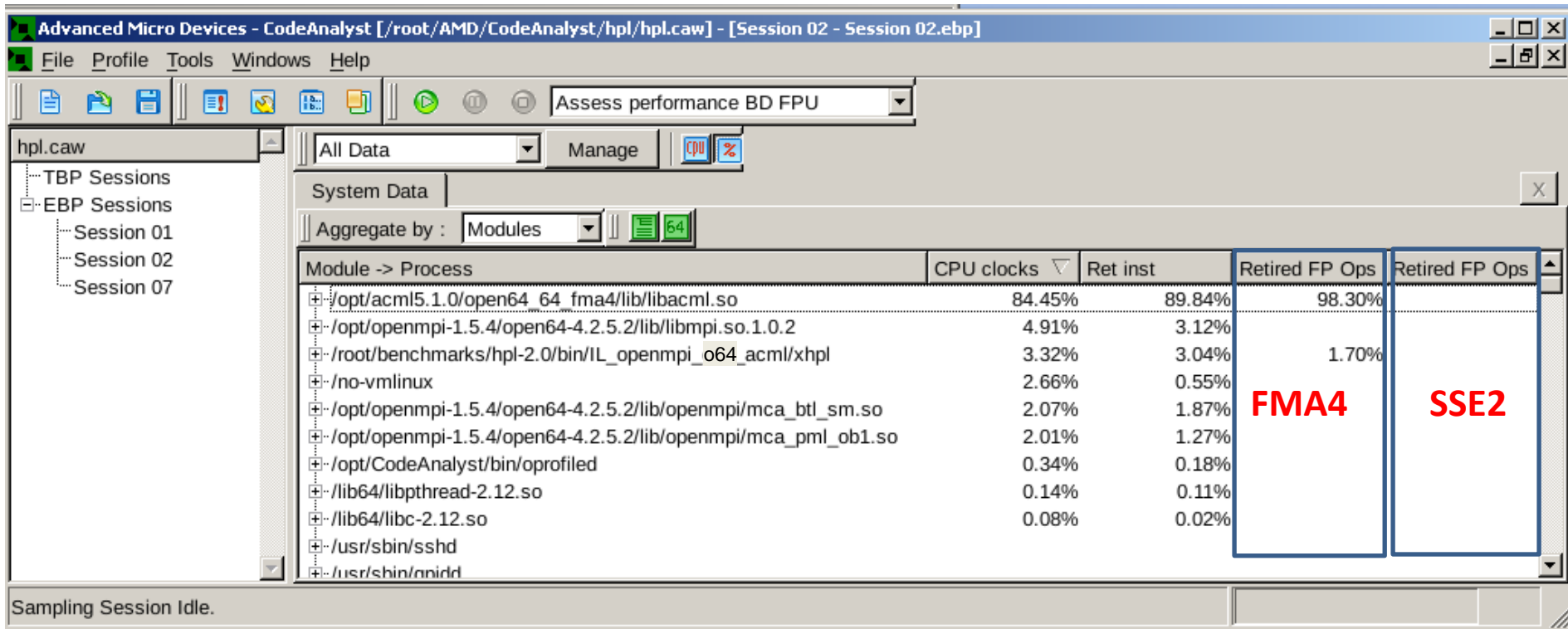


Testing BD FPU architecture with HPL (DGEMM)

Understanding/Profiling HPL workload:

- > 85% computational time spent in DGEMM.
- < 7% MPI communication overhead SHMEM or IB
- HPL efficiency \leftrightarrow DGEMM efficiency
- HPL power \leftrightarrow DGEMM power consumption
- DGEMM using ACML FMA4 version to issue
- 8 FLOPs/clock per Bulldozer module.
- equivalent to 4 FLOPs/clock per core.

CodeAnalyst session on HPL



- 84.5% clks at ACML library , 98.3% FPU ops are FMA4, 0% SSE2
- 4.9% + 2% + 2% ~7% clks at MPI

CodeAnalyst session on HPL (cont.)

Advanced Micro Devices - CodeAnalyst [/root/AMD/CodeAnalyst/hpl/hpl.caw] - [Session 02 - Session 02.ebp]

File Profile Tools Windows Help

Assess performance BD FPU

hpl.caw

All Data Manage CPU %

System Data ...opt/acml5.1.0/open64_64_fma4/lib/libacml.so - Data

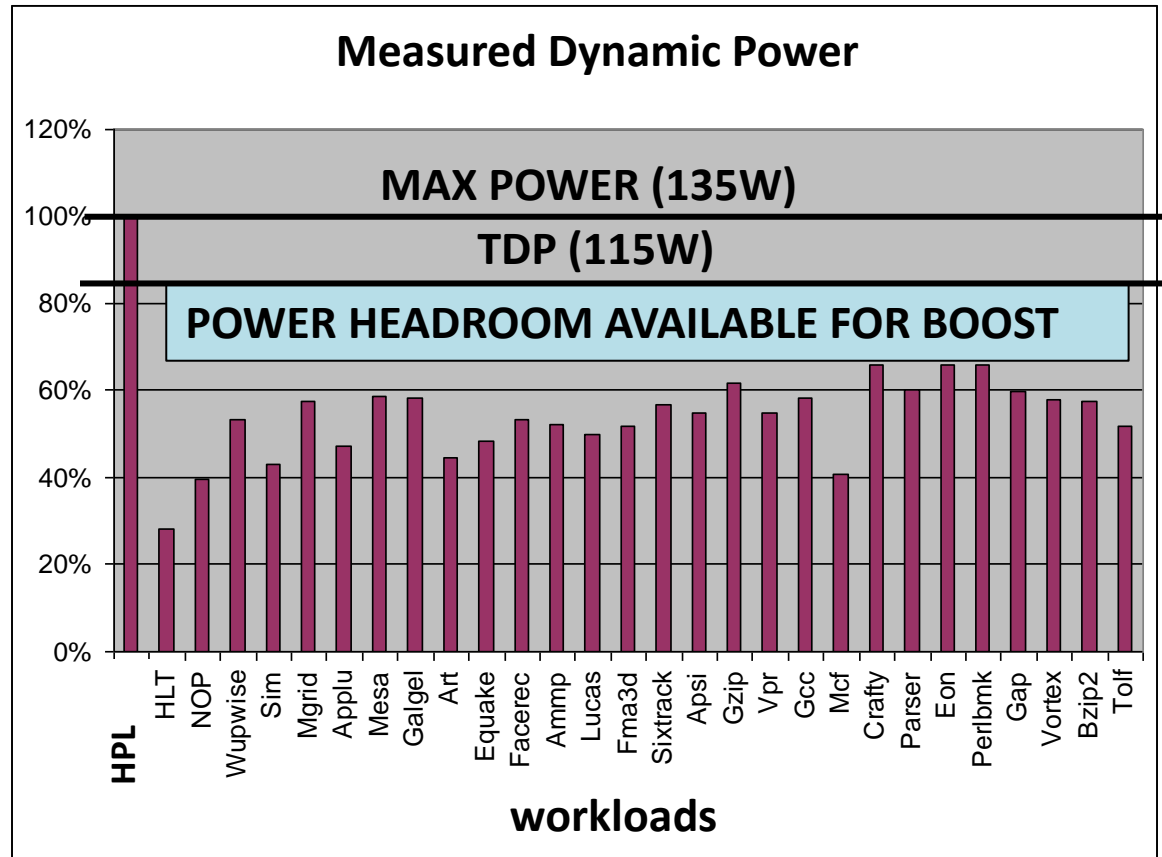
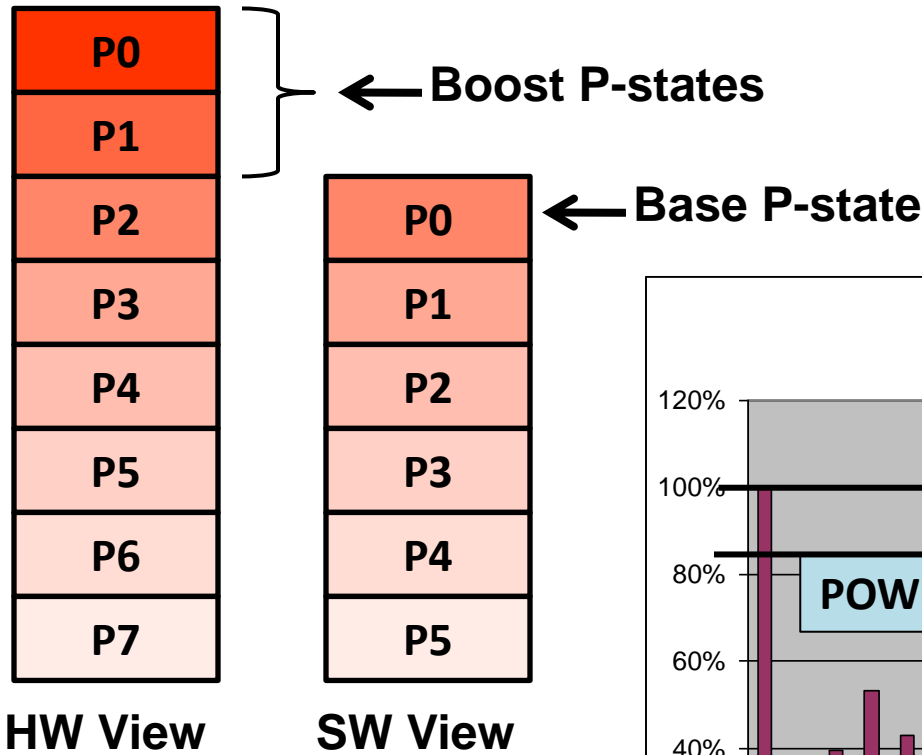
Pid : All Pid Tid : All Tid Show System Percentage

CS:EIP	Symbol + Offset	CPU clocks	Ret inst	Retired FP Ops	Retired FP Ops
0x8db080	dmmkernbd	82.15%	89.12%	94.81%	
0x8d7240	dmmavxbkka_	0.74%	0.10%		
0xb78d40	dvgemm_	0.58%	0.19%		
0xb5d140	dtrsmss_	0.57%	0.36%	2.12%	
0x8d6da0	dmmavxalphabkktb_	0.28%	0.03%	1.22%	
0x7e9300	dgemvnkernf_	0.09%	0.03%		
0xba20c0	idamax_	0.01%	0.02%		
0xaa9720	L1_128Bloop_srcdest_aligned	0.01%		0.14%	
0xbac7e0	ompaay_				
0xbac7c0	ompaax_				
0xbac700	ompaaw				

Sampling Session Idle.

- 82.1% cloks spent at dgemm kernel for Bulldozer architecture
- 94.8% FPU ops of dgemm kernel are of type FMA4, 0% SSE2

Advanced Power Management (ie. boost)



HPCMODE (no need to disable APM for HPL)

BIOS option or tool available at developer.amd.com

Without HPCMODE

Core Pstates :

Pb0 := Freq: 3200 MHz

Pb1 := Freq: 2600 MHz

P0 := Freq: 2300 MHz

P1 := Freq: 2100 MHz

P2 := Freq: 1800 MHz

P3 := Freq: 1600 MHz

P4 := Freq: 1400 MHz

With HPCMODE

Core Pstates :

Pb0 := Freq: 3200 MHz

Pb1 := Freq: 2600 MHz

P0 := Freq: 2300 MHz

P1 := Freq: 2300 MHz

P2 := Freq: 2300 MHz

P3 := Freq: 2300 MHz

P4 := Freq: 1400 MHz

Boost frequencies
not to exceed TDP

Sustained base
frequency for HPL
above TDP

Power savings, idle

HPL will work dithering between P0 and P3, consuming above TDP

HPL, 2P, 16cores @ 2.3GHz, 64GB, DDR3-1600

- SW stack: Open64 4.2.5.2, ACML 5.1.0, openMPI 1.5.4 , KNEM 0.98
- APM on + HPCmode off, 75.3% efficiency, 442W node (115W TDP)

```
=====
T/V          N  NB  P  Q          Time          Gflops
-----
WR01L4L2    86400 100  4  8          1937.96          2.219e+02
-----
Using acml fma4 single
threaded per core
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)= 0.0023599 ..... PASSED
=====
```

- APM on + HPCmode on, 82.3% efficiency, 475W node (135W MAXP)

```
=====
T/V          N  NB  P  Q          Time          Gflops
-----
WR01R2L2    86400 100  4  8          1774.55          2.423e+02
-----
Using acml fma4 single
threaded per core
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)= 0.0022068 ..... PASSED
=====
```

Detecting Thermal Throttling when running HPL

- Run as root across the cluster, script valid for Magnycours and Interlagos.

```
#!/bin/sh
host=$HOSTNAME
tmp1=a.$host
tmp2=b.$host
lspci -xxx -s 0:18.3 | grep "60:" > $tmp1
lspci -xxx -s 0:1a.3 | grep "60:" >> $tmp1
lspci -xxx -s 0:1c.3 | grep "60:" >> $tmp1
lspci -xxx -s 0:1e.3 | grep "60:" >> $tmp1
cat $tmp1 | awk '{print $6}' > $tmp2
sed 's/\(.*\)./\1/' $tmp2 > $tmp1
cat $tmp1 | awk '{print "echo \"ibase=16;obase=2; " $1 "\" | bc"}' > $tmp2
sh $tmp2 > $tmp1
rev $tmp1 > $tmp2
sed 's/\(.*\).../\1/' $tmp2 > $tmp1
sed -e 's/0/OK/g' -e 's/1/TT/g' $tmp1
rm $tmp1 $tmp2
```

Example:

```
[root@bdnode]# ./runtt.sh
```

```
OK
```

```
TT ← second processor
      is thermal throttling.
```

```
Processor operating at lower
frequency impacts on HPL score.
```

If TT:

```
Fix the cooling for that processor
and rerun HPL in order to achieve
good scores.
```

Thanks, Q&A

- **USEFUL FREE** resources at AMD website developer.amd.com:
 - Tools > Open64 compiler
 - Tools > CodeAnalyst
 - Libraries > ACML (AMD Core Math Library), AMDlibM
 - Libraries > ACML > hpcmode tool
 - Libraries > ACML > HPL binary
 - Docs> Dev. Guides > Compiler Options Quick Ref. Guide
 - Docs > Dev. Guides > SWOG (SoftWare Optimization Guide)
 - Docs > Developer Guides > Linux Tuning Guide
 - Docs > Articles & White Papers > HPC > HPL
 - Community > Developer Forums