1109 vs 1188 Floating Point Benchmark Results

Jan Pedersen
28 Aug. 1986

The timing results below compare the performance of an 1109 vs an 1188 on a suite of floating point benchmarks. The desire was to measure as closely as possible, using TIMEALL, the relative speeds of various arithmetic opcodes. No attempt was made to benchmark a "real" (e.g. linear algebra) application.

The 1109 was running a lispcore sysout(Makesysdate "21-Aug-86"), a real memory ize of 7167 pages, and a set of Weitek floating point chips.

The 1188 was running a lispcore sysout(Makesysdate "25-Aug-86"), a real memory size of 7424 pages, and no floating point hardware, but microcode support for serveral boxed and unboxed floating point opcodes.

Both boxed and unboxed opcodes were benchmarked. Most benchmarks were a tight loop with the opcode evaluated 10,000 times. The block floating point opcodes were evaluated 1,000 times on arrays of size 100 (for a total of 100,000 arithmetic operations). Some of the boxed opcodes produced no garbage since they returned one of their inputs as an output, or returned T or NIL.

The 1188 had no microcode support for the block opcodes (they ran in lisp using scalar unboxed opcodes).

Cpu time and GC time are recorded separately for the boxed opcodes. Cpu time and CPU less the CPU time for an empty loop are recorded separately for the unboxed opcodes.

NA stands for Not Applicable.

Boxed Float Results (Time in seconds)

	1109		1188		Ratio	
Opcode	Cpu	Gc	Cpu	Gc	Cpu	Gc
FPLUS	.98	(2.35)	1.02	(2.05)	.96	(1.15)
FDIFF	.98	(2.35)	1.03	(2.05)	.96	(1.15)
FTIMES	.99	(2.35)	1.17	(2.05)	.85	(1.15)
FQUOT	1.36	(2.34)	1.19	(2.05)	1.14	(1.15)
FGREATP	.304	(0.0)	.267	(0.0)	1.14	(NA)

	1109		1188		Ratio	
Function	Cpu	Gc	Cpu	Gc	Cpu	Gc
FABS	2.1	(2.33)	2.14	(2.03)	.98	(1.15)
FMINUS	1.13	(2.33)	1.11	(2.04)	1.02	(1.14)
FIX	6.56	(0.0)	5.85	(0.0)	1.12	(NA)
FMAX	1.15	(0.0)	1.04	(0.0)	1.10	(NA)
FMIN	1.14	(0.0)	1.02	(0.0)	1.12	(NA)

Unboxed Float Results (Time in seconds)

	1109		1188		Ratio	
Opcode	Cpu	(- empty)	Cpu	(- empty)	Cpu	(- empty)
Empty lp	.109	(NA)	.097	(NA)	1.12	(NA)
UFPLUS	.244	(.135)	.363	(.266)	.67	(.508)
UFDIFF	.244	(.135)	.362	(.265)	.67	(.509)
UFTIMES	.26	(.151)	.515	(.418)	.50	(.361)
UFQUOT	.616	(.507)	.533	(.436)	1.16	(1.16)
UFGREATP	.235	(.126)	.206	(.109)	1.14	(1.16)
UFABS	.178	(.069)	.161	(.064)	1.11	(1.08)
UFMINUS	.179	(.07)	.161	(.064)	1.11	(1.09)
UFIX	.213	(.104)	.205	(.108)	1.04	(.963)
UFMAX	.235	(.126)	.206	(.109)	1.14	(1.16)
UFMIN	.231	(.122)	.206	(.109)	1.12	(1.12)
BLKPLUS	.39	(.281)	6.73	(6.63)	.058	(.042)
BLKDIFF	.384	(.275)	5.71	(5.61)	.067	(.049)
BLKTIMES	.39	(.281)	7.63	(7.53)	.051	(.037)
POLY	.45	(.341)	4.92	(4.82)	.091	(.071)

Summary

a.) Unboxed operations are a factor of ten faster than boxed operations across the board.

b.) On an 1109 the block opcodes yield another factor of five to ten.

c.) For scalar operations, the 1188 is never worse than .36% of the 1109, and never better than 1.16% of the 1109.

d.) The 1188 was actually faster than the 1109 for several unboxed opcodes -- and generally faster for the boxed opcodes.

e.) The 1109's floating point hardware really comes to the fore in the block opcodes. Unfortunately, with the exception of polynomial opcode, these opcodes are rarely used.

J.P.