AMD SIMD extensions

Home
SIMD extensions
HOWTO
Intel SIMD extensions
AMD SIMD extensions
Example programs
Speed test:
Telecom application
...in C#
...in Perl
...in Python
Result & conclusion
Need for things
..Xilinx CPLD
...JTAG programmer
...AVR microcontroller
...self-made PCI board
...D.A.L. calculator
...a good printer
...a Z32 computer
...8-bit will die
Low voltage
..squeeze it out from 1.2V
A bit about 845GE
Contact

MMX

- for 64-bit integer SIMD operations
- fully compatible with Intel
- 57 instructions and 4 datatypes
- shares registers with x86 FPU
- up to two MMX instructions per clock

3DNow!

- 19 SIMD floating-point instructions
- two special instruction for prefetching and faster entry/exit for the MMX state
- shares registers with MMX instruction set
- MMX/3DNow! code can be freely mixed:
  in other words:
  - you have SIMD floating-point instructions if you are in MMX state
  - is it as AMD just extended MMX with floating-point and did it even further with
  3DNow! Professional
- handles up to two 32-bit single precision floating-point numbers in 64-bit register
- up to two 3DNow! instructions per clock through two register executions pipeline (total of four floating-point instruction)
- no penalty for switching between x86 FPU and MMX/3DNow! instructions
- 3DNow! provide for a flat register file (normal x86 is stack-based) and this simplifies programming
- it is recommended to use 3DNow! instead of x86 instructions

3DNow! Professional

- adds 24 new instructions to MMX/3DNow!
- shuffle, streaming, packed values minimum/maximum, byte mask move
- Intel SSE 1 instructions
- FPU can handle three operations of any mixture of x86/MMX/3DNow! instruction per cycle
- it is still needed to clean register file with FEMMS/EMMS when switching between x86 and MMX/3DNow!

Athlon 64 and Opteron

- the full set of MMX, 3DNow!, SSE and SSE2
- SSE and SSE2 instructions work well in both of 32-bit and 64-bit programs
- additional 8 XMM + 8 general-purpose registers (GPRs) in 64-bit mode
- for SIMD operations of integer and floating-point the SSE and SSE2 should be used
- when writing 32-bit code it is better to avoid x87 and 3DNow! floating-point instructions
  and use only SSE/SSE2 if you want to be able to easily migrate to 64-bit code in the future

Comments

It is unbelievable how much more powerful the AMD microarchitecture is comparing to Intel.
In fact when comparing lowend AMD Duron 1.6 GHz (128KB+64KB cache) to Intel Pentium 4 1.8A (512KB cache) then Duron is winner in many of benchmarks (www.anandtech.com)!
Things are even getting worse for Intel when we try to compare Pentium 4 based Celerons with Duron. Unbelievable - nowadays 1 GHz can mean nothing. 1.6 GHz Duron from AMD is at least in the same performance level as 2.6 GHz Celeron from Intel and even wins it. Small cache is not suitable for Pentium 4 architecture. Take it or leave it.

Last update: *25-jul-2004*
indrek.kruusa : tuleriit dot ee