Characterization of Native Signal Processing Extensions

Jason Law
EE382C Embedded Software Systems
Spring 2002
The Problem Statement

• Intel’s MMX and SSE are fairly well understood

• Intel developed new extensions – SSE2
  – 144 new instructions, widen MMX registers to 128 bit
  – Includes data prefetch instructions

• How well does SSE2 work?
  – Is it easy to write code that efficiently uses SSE2?
  – Is there a speedup over SSE and MMX?
Native Signal Processing

• NSP extensions incorporate SIMD into older architectures
  – SIMD takes advantage of data level parallelism
  – One SIMD instruction can do the work of many scalar instructions
  – Significant speedup seen for many DSP type algorithms
    • FFT, FIR filters, matrix math, convolutions, etc.

• Extending an architecture for NSP causes several problems
  – Compilers are not very good at utilizing latest features
  – SIMD does not help some algorithms
  – Extensions vary greatly across architectures and manufacturers
Solution

• Several benchmark suites exist that I pulled from:
  – Mediabench, MiBench, BDTI’s suite, and EEMBC’s suite
  – MiBench runs under Linux, need a good API to measure

• Test C and assembly kernels and multimedia applications
  – FFT, FIR filters, MPEG encoders/decoders, MP3 players, etc.
  – Applications can be run under Windows and measured with Vtune

• Setup requires root/administrator access
  – Use Vtune or API to record hardware performance monitor data and then analyze offline
    • Look at data prefetch characteristics
    • Execution time, number of SSE2 instructions, etc.

• Models of computation:
  – Kernels can be modeled as SDF or BDF
  – Applications can be implemented as process networks
Results

• Linux performance monitoring tools have problems with Pentium 4!
• MiBench successfully runs
  – Need to instrument and analyze
• Kernels are written
  – Need to instrument and analyze
• Applications in Windows have been selected
  – Quake 3 Demo
  – Flask DivX video encoder/decoder
  – Canon .wav to .mp3 converter
Conclusions

- Performance monitoring API’s need to be updated to work with Pentium 4 and Linux
  - Currently the only reliable method I have found is to use Vtune and Windows
  - Pentium 4 has 18 counters, Pentium 3 has 2 counters
- Code generation easiest with Intel C/C++ compiler
  - This works well for both Windows and Linux, at least as far as I can tell
- Deliverables
  - Investigation of speedup of SSE2 over SSE and MMX