Characterization of Native Signal Processing Extensions

Jason Law

EE382C Embedded Software Systems

Spring 2002
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
The Problem to Tackle

• 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
• Is there a speedup over SSE and MMX?
  – Does data prefetch make a difference?
• Want to avoid comparing only to other Intel systems
  – Interesting to compare to AMD’s Enhanced 3DNow!
  – Interesting to compare to PowerPC’s Altivec
Plans and Goals

- Several benchmark suites exist that I can pull from:
  - Mediabench, MiBench, BDTI’s suite, and EEMBC’s suite
  - I have access to code that previous work used for MMX and SSE
- Test C and assembly kernels and multimedia applications
  - FFT, FIR filters, MPEG encoders/decoders, MP3 players, etc.
- Use Vtune or other similar tools to record hardware performance monitor data and then analyze offline
  - Look at data prefetch characteristics
  - Execution time, number of instructions, etc.
- Models of computation:
  - Kernels can be modeled as SDF or BDF
  - Applications can be implemented as process networks