Processor Design
–
Benchmarking Processor Performance

Professor Jari Nurmi
Institute of Digital and Computer Systems
Tampere University of Technology, Finland
email jari.nurmi@tut.fi

Factors of Performance

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	instr count	CPI	clock rate
Program	X		
Compiler	X	X	
Instr. Set	X	X	
Organization		X	X
Technology			X

Performance Analysis and Benchmarks

- Time is the measure of computer performance!
- Remember Amdahl's Law:
  - Speedup is limited by unimproved part of program
- Good products created when have:
  - Good benchmarks
  - Good ways to summarize performance
- If NOT good benchmarks and summary, then choice between
  1) improving product for real programs
  2) changing product to get more sales (sales almost always wins)

Performance Analysis and Benchmarks (Cont'd)

Pros		Cons
• representative	Actual Target Workload	• very specific • non-portable • difficult to run, or measure • hard to identify cause
• portable • widely used • improvements useful in reality	Full Application Benchmarks	•less representative
• easy to run, early in design cycle	Small "Kernel" Benchmarks	• easy to "fool"
• identify peak capability and potential bottlenecks	Microbenchmarks	• "peak" may be a long way from application performance

Performance Analysis and Benchmarks (Cont'd)

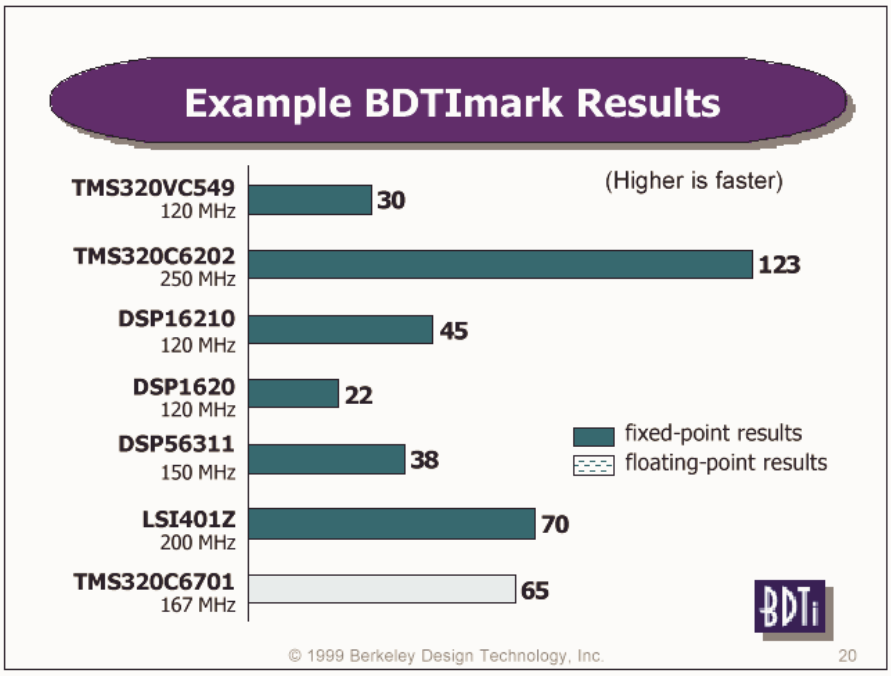
- SPEC95
  - For general purpose processors/workstations
  - Eighteen application benchmarks (with inputs) reflecting a technical computing workload
  - Eight integer
    - go, m88ksim, gcc, compress, li, ijpeg, perl, vortex
  - Ten floating-point intensive
    - tomcatv, swim, su2cor, hydro2d, mgrid, applu, turb3d, apsi, fppp, wave5
  - Must run with standard compiler flags
    - eliminate special undocumented incantations that may not even generate working code for real programs

DSP Benchmarks

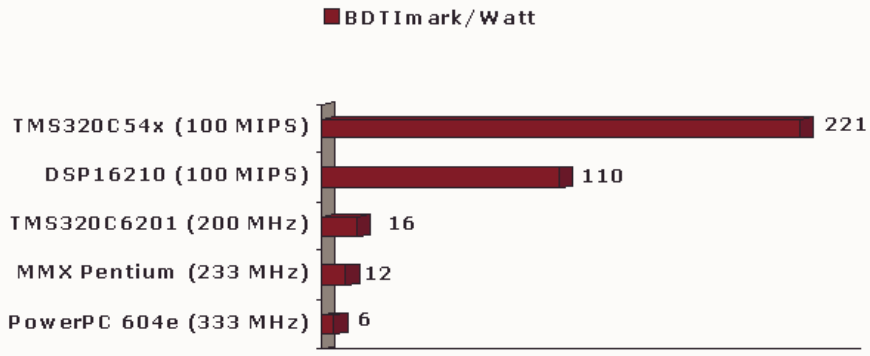
- BDTImark
  - Berkeley Design Technology, Inc. proprietary
  - algorithm kernel based benchmark
  - derived from important DSP applications
  - implemented in a consistent fashion
  - carefully hand-optimized for each processor
  - verified by an independent third party (BDTI)
  - in addition to BDTImarks (a composite speed metric), also
    - BDTImark/Watt
    - BDTImark/$

DSP Benchmarks (Cont'd)

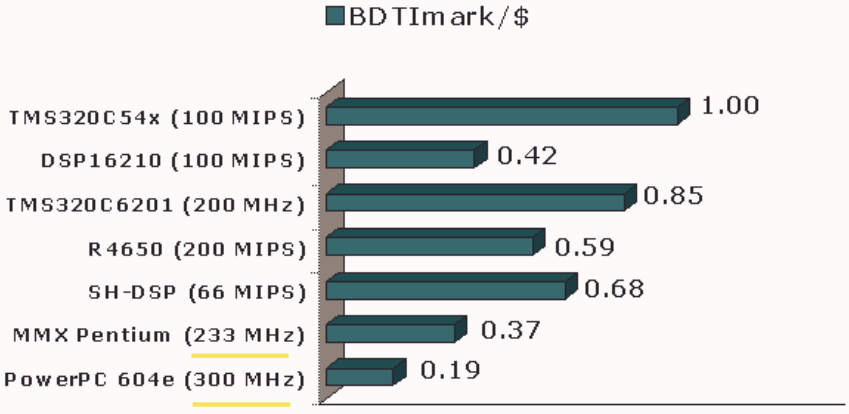
- BDTImark includes the following benchmark kernels
  - real block FIR
  - complex block FIR
  - real single-sample FIR
  - LMS adaptive FIR
  - IIR
  - vector dot product
  - vector add
  - vector maximum
  - convolutional encoder
  - finite state machine
  - 256-point radix-2 in-place FFT



Benchmark Results: Energy Efficiency



Benchmark Results: Cost-Perf.



DSP Benchmarks (Cont'd)

- DSPstone
  - RWTH Aachen (prof. Heinrich Meyr)
  - consists of **three** benchmark suites of different granularity
    - application benchmarks
    - DSP-kernel benchmarks
    - C-kernel benchmarks
  - tries to measure separately hardware and C-compiler by having
    - a reference code R (hand-coded) and
    - generated code G
  - compiler efficiency metrics include the following *reference code distances*
    - execution time overhead
    - clock-cycles overhead
    - program memory overhead
    - data memory overhead
    - memory overhead

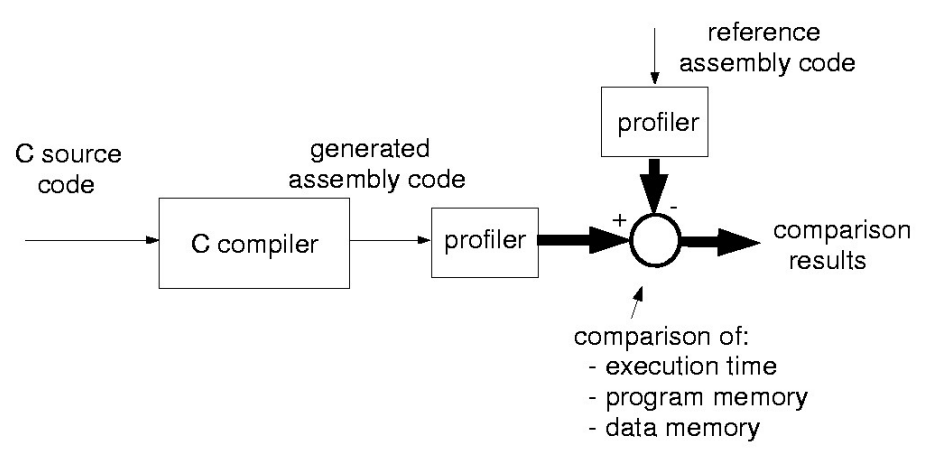
DSP Benchmarks (Cont'd)

- DSPstone DSP-Kernel benchmarks are
  - real updates
  - complex updates
  - matrix product
  - convolution
  - complex product
  - IIR biquad section
  - IIR filter
  - FIR filter
  - LMS filter
  - FFT
  - 2-D filter
- Reference codes to these are included in the DSP development packages, the C-programs are developed by RWTH Aachen

DSP Benchmarks (Cont'd)

- DSPstone HLL-Kernel benchmarks are
  - CALL – C function call overhead
  - FOR – for loop analysis
  - NESTED-FOR – nested for loop analysis
  - DOWHILE – do while loop analysis
  - WHILE – while loop analysis
  - FLOAT – floating-point arithmetic performance
  - INT – integer arithmetic performance
  - FRACT – fractional arithmetic performance
  - LONGINT – long int arithmetic performance
  - MADD – usage of multiply-add instruction
  - PARALLEL – instruction parallelism analysis
  - INDEXING – indexing vs. pointer addressing
  - COMPACTION – effects of source code compaction
- Benchmarks used also to find out the forms of the C code which are best suited for the particular compiler

DSPstone Principle



DSPstone Application Benchmark Example

	AD 2101	Motorola 56001	TI C51
cycles $\Delta c[\%]$	698	510	555
program memory $\Delta p[\%]$	284	51	7
data memory $\Delta d[\%]$	383	175	301
memory $\Delta m[\%]$	302	70	30

ADPCM Benchmark: Compiler Overhead.

Embedded Processor Benchmarks

- EDN Embedded Microprocessor Benchmark Consortium (EEMBC - pronounced embassy)
  - hybrid of real-world and synthetic benchmarks
  - targets the automotive/industrial, consumer, networking, office automation, and telecommunications markets, including
    - engine control
    - digital cameras
    - printers
    - cellular phones
    - modems
    - etc.
  - 37 individual algorithms
  - both "out-of-the-box" and optimized benchmark scores

Summary

- Several benchmark suites and other measures available
- Different levels of granularity and real-worldness
- Different application domains
  - general purpose computing
  - DSP
  - embedded applications
- In embedded and DSP, also cost-performance ratio, performance per (milli)watt and memory footprint of programs are important measures
- Compiler performance may/may not have a major impact on the score – or it is maybe measured separately
- Performance in your key application is the bottom-line, necessarily none of the benchmarks will indicate that!

End of Benchmarking