Multicore Benchmarks Help Match Programming to Processor Architecture

Shay Gal-On
Director of Software Engineering
EEMBC
www.eembc.org
What’s Up?

• What is EEMBC
• Multicore benchmarking framework
• Multicore sample results
Our Focus

• Industry standard benchmarks since 1997

• New: Evaluating current and future development of MP platforms
  • Uncovering MP bottlenecks
  • Analysis of multicore systems
MultiBench – Overview

• Multicore is everywhere
  • Hence our focus on benchmarking embedded multicore solutions.

• Initial implementation targets SMP
  • Easiest model to work with
  • Current metrics misleading (rate, DMIPS, etc)

• Workloads and work items
  • Develop workloads closer to real life
  Individual kernels are work items within a workload
MultiBench – Abstraction

- Easy to run
  - Only 13 calls to implement means quick porting to any platform/OS/toolchain
- Wide range of applications
  - Sufficient functionality with minimum porting effort for most embedded applications.
- Most EEMBC benchmarks ported
  - Thread safety
  - New datasets
  - Common API
Workloads and Work Items

- Multiple algorithms
- Multiple data
- Decomposition

Multiple contexts within an item

Workload

- Work Item $A_0$
- Work Item $A_1$
- Work Item $B_0$
Concurrency

Multiple Work Items
Note: Algorithm same, different datasets

Can be run one item at a time ......

Or concurrently, depending on processing resources and scheduling
Multiple Iterations
Example with two contexts doing four repeats

NOTE:
Alternatively, the OS may schedule Core 1 and Core 2 to run only hamburgers and cucumbers, respectively. Other combinations are possible.
Work Items and Workers

A collection of threads working on the same item are referred to as workers.
Workload set #1

• Carefully selected subset
  • Only 30 workloads over a few kernels
• Benchmark efficiency of various multi-processing system related effects
  • Computation / Synchronization
  • Memory / Cache
• Work items from multiple segments
  • Networking
  • Consumer
Workloads scale in mysterious ways...

Huge drop in performance when oversubscribed

Nice scaling on networking only workloads

Some benchmarks plateau earlier than expected
Various workloads expose different trends in systems.
Image Rotation, 90° CW

• Simple algorithm, very little computation
• Easy to slice to workers, since each pixel is independent.
• Depending on slicing and image size, can exercise the system in interesting ways.
• Many other applications use similar data movement patterns.
Rotation Benchmark (Slice = 1)

Peak performance with 16 active threads (~11x speedup)

Worth noting that for less than 8 workers, peak performance is achieved with utilization of 1/2 total cores or less.
Rotation Benchmark (Slice = 4)

Peak performance at 2.14. 
8*4 = 32 active threads 
~11x over base

Also interesting to note that peak performance for any number of workers is achieved with 4 images being handled concurrently
Rotation Benchmark (Slice = 32)

With larger slices, performance actually drops, with peak performance at 1.27. ~7x over base.

Worth noting performance drops even with 1 worker past 4 concurrent items.
Use results to design software!

• Results show low overhead for sync
  • Take advantage of data decomposition.
  • Use medium to small granularity.

• Results show throughput max with small number of concurrent data streams
  • Pay more attention to the lower level concurrency.

• Results show bottlenecks
  • Make sure system resources are not oversubscribed by testing with lower load.
Performance Comparison, Rotation, 2x2G vs 16x500MHz

Significant drop in performance on the dual core machine when using more than one worker.

This is mostly due to cache coherency when writing the rotated image.

The 16 core machine OTOH scales nicely.

16 core wins on absolute performance
So which system is better?

Comparing max performance of powerful 2x2G with simple 16x500MHz

Depends, what are you using it for?
Composing Workloads
EEMBC Benchmarking Services

• Analysis of complex systems
  • Take advantage of in-house expertise
  • Quickly pick the right processor

• Targeted benchmarking
  • Creation of relevant benchmarks
  • Creation of automated benchmark systems

• Quick benchmark results
  • Utilize expertise with multiple embedded environments
  • Utilize experience with EEMBC suites and other benchmarks

• Guidance for optimizing your applications
Summary

• EEMBC has a new suite for burning multicore issue.

• Use EEMBC benchmarks to guide software development as well as analyze platforms.

• EEMBC launched new service to assist with benchmarking and analysis.
Questions?